Open-Source-Projekte als Utopie, Methode und Innovationsstrategie: historische Entwicklung - sozioökonomische Kontexte - Typologie

Schrape, Jan-Felix

Veröffentlichungsversion / Published Version
Monographie / monograph

Jan-Felix Schrape

Open-Source-Projekte als Utopie, Methode und Innovationsstrategie

Historische Entwicklung — sozioökonomische Kontexte — Typologie

vwh
Verlag Werner Hülsbusch
Fachverlag für Medientechnik und -wirtschaft

J.-F. Schrape: Open-Source-Projekte als Utopie, Methode und Innovationsstrategie

Überarbeitete und erweiterte Fassung des Diskussionspapiers
„Open Source Softwareprojekte zwischen Passion und Kalkül“
(Stuttgarter Beiträge zur Organisations- und Innovationsforschung 2015-02)

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet unter http://d-nb.de abrufbar.

vwh Verlag Werner Hülsbusch
Fachverlag für Medientechnik und -wirtschaft

www.vwh-verlag.de

Korrektorat und Satz: Werner Hülsbusch
Umschlag: design of media, Lüchow
Druck und Bindung: SOWA Sp. z o. o., Piaseczno

Printed in Poland

ISBN: 978-3-86488-089-6

Inhaltsverzeichnis

1 Einleitung

Quelloffene Softwarekomponenten spielen für die basalen Infrastrukturen des Internets eine zentrale Rolle: Der *Apache HTTP Server* ist seit 1996 die meistverbreitete Webserver-Software; *Linux*-basierte Architekturen dominieren den Markt für Server-Betriebssysteme; der Großteil der im Netz eingesetzten Content-Management- und Datenbanksysteme steht unter freien Lizenzen (vgl. Netcraft 2015; W3techs 2015). Das Marktforschungsunternehmen *Gartner* beobachtet eine vermehrte Nutzung von Open-Source-Software in Organisationen und geht davon aus, dass 2016 die Mehrheit aller IT-affinen Firmen auch in kritischen Bereichen auf quelloffene Elemente zurückgreifen wird (vgl. Driver 2014). Und auf dem Feld der Consumer-Software sind Produkte, die vollständig oder teilweise auf Open-Source-Projekten basieren, zu einem festen Bestandteil des Alltags vieler Anwender geworden, so etwa der Browser *Mozilla Firefox* oder das mobile Betriebssystem *Android*. Auch wenn sich eines der vorrangigen Ziele des ‚free software movements' – die Ablösung von *Microsoft Windows* durch *GNU/Linux* auf dem Desktop – gemessen an den globalen Verbreitungszahlen für 2015 (*Windows*: 91 Prozent; *Mac OS X*: 7 Prozent; *Linux*: unter 2 Prozent) bislang nicht erfüllt hat (vgl. NetApplications 2015), kann quelloffener Software heute in vielen Segmenten des IT-Marktes eine hohe Relevanz zugeschrieben werden.

Diese zunehmende Bedeutung von Open-Source-Projekten in der Softwareentwicklung wurde in den Sozialwissenschaften angesichts klassischer Sichtweisen, die ‚intellectual property rights' als wesentliche Treiber in Innovationsprozessen ansehen (vgl. z. B. Arrow 1962; Romer 1990), zunächst mit Erstaunen zur Kenntnis genommen (vgl. Lessig 1999) und nachfolgend von einigen Autoren als Beleg für die Emergenz eines neuen Produktionsmodells formatiert, das auf freiwilliger wie selbstgesteuerter Kollaboration beruht, den Stellenwert korporativer Akteure in der Arbeitswelt schmälert und eingespielten Formen ökonomischer Koordination auf Dauer überlegen sein könnte (vgl. Lakhani/Hippel 2003; Osterloh/Rota 2007; Suddaby 2013). Insbesondere die durch Yochai Benkler popularisierte Vorstellung der ‚commons-based peer production' als technisch effektivierte "collaboration among large groups of individuals [...] without relying on either market pricing or managerial hierarchies to coordinate their common enterprise" (Benkler/Nissenbaum 2006: 394), die mit "systematic advantages [...] in identifying and allo-

cating human capital/creativity" (Benkler 2002: 381) einhergehen soll, erfuhr eine intensive Reflexion und Anwendung auf angrenzende Kontexte (vgl. z. B. Kostakis/Papachristou 2014; Rifkin 2014; Baldwin/Hippel 2011).

Zweifelsohne lassen sich unter den inzwischen millionenfach initiierten Open-Source-Projekten (vgl. Black Duck/North Bridge 2015) Entwicklergemeinschaften finden, die sich durch mehr oder minder ausgeprägte Muster einer "radically decentralized, collaborative, and nonproprietary [...] peer production" (Benkler 2006: 60) auszeichnen und deren Teilhabende sich primär aus intrinsischen Motiven – „satisfying psychological needs, pleasure, and a sense of social belonging" (Benkler 2004: 1110) – oder indirekten extrinsischen Anreizen (z. B. zur Steigerung der persönlichen Reputation) dort einbringen. Empirische Untersuchungen legen allerdings nahe, dass viele marktrelevante Open-Source-Projekte mittlerweile in erster Linie durch die Beiträge von in etablierten IT-Unternehmen angestellten Softwareentwicklern getragen werden. In der oft als typisches Beispiel herangezogenen *Linux Kernel*-Community etwa wurden laut einer Erhebung der *Linux Foundation* (vgl. Corbet/Kroah-Hartman/McPherson 2015: 11) zuletzt über 80 Prozent der Entwicklung von Programmierern durchgeführt, "who are being paid for their work" – unter anderem von *Google*, *IBM* und *Intel*, die mit Blick auf ihre eigenen Geschäftsaktivitäten an der Weiterentwicklung des Kernels interessiert sind. Darüber hinaus stehen viele der dauerhaft aktiven Open-Source-Vorhaben in einem engen Verweisungszusammenhang mit kommerziellen Angeboten oder werden über Spenden an ihre Dachstiftungen durch führende Technologiekonzerne mitfinanziert.

In Anbetracht dieser Verschränkungen von quelloffener und proprietärer Softwareentwicklung, deren unternehmerische Vor- und Nachteile in der Managementforschung unter Stichworten wie ‚open innovation' (vgl. West et al. 2014) diskutiert werden, reichen die in den Sozialwissenschaften nach wie vor oft üblichen, eher pauschalen Verweise auf Open-Source-Communitys als Alternative oder als Gegenentwurf zur kommerziellen Softwareentwicklung nicht mehr aus, um dem breiten Spektrum an unterschiedlich ausgerichteten Projekten in diesem Bereich gerecht zu werden. Dies zeigt sich auch in den Deutungskonflikten, die sich entlang von Begriffen wie ‚Free Software', ‚Open Source Software' und ‚Free/Libre Open Source Software' entsponnen haben, sowie in den Ausrichtungskontroversen zwischen den zwei großen Bezugsorganisationen für quelloffene Software – der pragmatisch orientierten *Open Source Initiative* und der gesellschaftspolitisch fundierten *Free Software Foundation*.

Das vorliegende Buch verfolgt daher das Ziel, auf der Grundlage von aggregierten Marktdaten, Dokumentenanalysen, Literaturauswertungen sowie Hintergrundgesprächen mit Softwareentwicklern einen bündelnden sowie systematisierenden Überblick über Open-Source-Communitys und ihre sozioökonomischen Kontexte zu entfalten. Zunächst erfolgt eine historische Rekonstruktion zur Herausbildung quelloffener Softwareprojekte, die aufzeigt, dass freie und proprietäre Entwicklung seit jeher ineinandergreifen. Nachfolgend werden die Beziehungen zwischen Open-Source-Communitys und etablierten IT-Unternehmen diskutiert. Daran anknüpfend werden vier typische Varianten aktueller Open-Source-Projekte voneinander abgegrenzt – von korporativ geführten Kollaborationsprojekten und elitezentrierten Projektgemeinschaften über heterarchisch angelegte Infrastrukturvorhaben bis hin zu egalitär ausgerichteten Entwicklergruppen, die am ehesten der Idee einer ‚commons-based peer production' entsprechen. Abschließend werden die sich wandelnden Relationen zwischen freier Software und allgemeinem Markt in verdichteter Form nachgezeichnet: Während quelloffene Softwareentwicklung zunächst subversiv konnotiert war und in geschützten bzw. vom allgemeinen Markt abgekoppelten Nischen stattfand, ist das Involvement in Open-Source-Projekte heute zu einem festen Bestandteil der Innovationsstrategien aller großen Anbieter geworden.

2 Historische Entwicklung

Dass Unternehmen ihre Wissensbestände insbesondere zu Beginn von Innovationsprozessen auch außerhalb formaler Kooperationsbeziehungen miteinander teilen, ist kein neuartiges Phänomen und wurde bereits durch Robert C. Allen als weithin unerforschter Prozess der *collective invention* beschrieben:

> "To the degree that economists have considered this behaviour at all, it has been regarded as an undesired 'leakage' that reduces the incentives to invent. That firms desire such behaviour and that it increases the rate of invention by allowing cumulative advance are possibilities not yet explored." (Allen 1983: 21)

Tabelle 1: Historische Beispiele für ‚collective invention'

Episode	Austauschprozesse	Resultat
The Cornish Pumping Engine ca. 1810–1850, Cornwall/London, England (vgl. Nuvolari 2004)	Austausch technischer Daten; Vergleich der individuellen Fortschritte über Fachjournale	Entwicklung einer brennstoffsparenden Hochdruck-Dampfmaschine für den Bergbau
Papierherstellung ca. 1827–1857, New England, USA (vgl. McGaw 1987)	stabile Gemeinschaft von Mühlenbesitzern; regelmäßiger informeller Erfahrungsaustauch	Erhöhung der Produktivität durch zunehmende Mechanisierung der Produktion
Hochofentechnologie ca. 1850–1880, Cleveland District, England (vgl. Allen 1983)	Wissensaustausch über Fachgesellschaften und Journale; kollektiver Trial- & Error-Prozess	Verringerung der Energiezufuhr durch Steigerung der Bauhöhen und Temperaturanpassungen
Flachbildschirme (LCD, Plasma) ca. 1969–1989, Japan/Europa/Nordamerika (vgl. Spencer 2003)	wissenschaftliche Publikation von Ergebnissen aus firmeneigener Forschung	inkrementelle Verbesserung und Technologieentwicklung in der vorkommerziellen Phase
Homebrew Computer Club ca. 1975–1978 [1986], Silicon Valley, USA (vgl. Meyer 2003)	Austausch technischer Informationen bis zu den ersten Markterfolgen ausgegründeter Firmen	Entwicklung erster marktfähiger persönlicher Mikrocomputer und anderer IT-Produkte

Von den von Robert C. Allen und Kollegen diskutierten historischen Beispielen (s. Tab. 1) hebt sich das Open-Source-Modell allerdings durch zwei entscheidende Merkmale ab (vgl. Cowan/Jonard 2003; Osterloh/Rota 2007): Zum einen unterliegt der Austausch in onlinebasierten Arbeitsgemeinschaften keiner geografischen Begrenzung mehr und bleibt nicht auf spezifische Branchenkontexte beschränkt. Zum anderen haben neuartige lizenzrechtliche Konstruktionen wie die *GNU General Public License*, die absichert, dass auch Derivate freier Software ausschließlich unter den gleichen Bedingungen weitergegeben werden können (‚Copyleft'), sowie die im Web erleichterte Durchsetzung informeller Regeln und Arbeitskonventionen dazu beigetragen, dass Open-Source-Softwareprojekte im Gegensatz zu früheren Ausprägungen kollektiver Invention auch nach der Herausbildung eines dominanten Designs und dessen kommerzieller Verwertung überlebensfähig bleiben.

Vor diesem Hintergrund lässt sich die Geschichte der quelloffenen Softwareentwicklung in vier Stadien einteilen:

(1) In den 1950er-Jahren, als Software noch nicht als separates Produkt galt, sondern zusammen mit der Hardware vertrieben wurde, folgte die Zusammenarbeit von universitären und korporativen Forschern freien akademischen Prinzipien (*kollektive Invention*).

(2) Ab Mitte der 1960er-Jahre bildete sich eine eigenständige Softwareindustrie heraus und der Anteil restriktiv lizenzierter Software stieg an. Gleichzeitig entstanden frühe Interessengruppen von Computer-Hobbyisten (*Kommodifizierung – Subkulturgenese*).

(3) Die elektronische Vernetzung förderte ab 1980 die ortsunabhängige Verdichtung freier Entwicklergemeinschaften, die ab 1985 durch die *Free Software Foundation* unterstützt wurden. Daneben wurden erste ‚Copyleft'-Lizenzen spezifiziert (*Institutionalisierung*).

(4) Um freie Software von ihren ideologischen Konnotationen zu lösen, schufen Szeneprotagonisten ab 1998 das Label ‚Open Source' und gründeten die *Open Source Initiative*. Die durch sie zertifizierten flexibleren Lizenzen trugen dazu bei, die sozioökonomische Relevanz quelloffener Software weiter zu steigern (*Wachstum – Diversifizierung*).

2.1 Kollektive Invention (1950er-Jahre)

Die ersten in Serie hergestellten digitalen Computer wurden in enger Kooperation zwischen staatlichen bzw. universitären Organisationen und privatwirtschaftlichen Unternehmen entwickelt, die ihr Geld in erster Linie mit der Produktion der kostspieligen Hardware verdienten. Der ab 1951 vorrangig in administrativen Einrichtungen installierte Großrechner *UNIVAC I* etwa schlug mit bis zu 1,5 Mio. US-Dollar zu Buche; die ab 1953 rund zweitausendmal verkaufte *IBM 650 Magnetic Drum Data-Processing Machine* kostete 1959 noch ca. 150.000 US-Dollar (vgl. IBM 1959; Ceruzzi 1998). Frühe Computerprogramme wurden mit Source-Code (den für den Menschen lesbaren Text eines Programms) weitergegeben, um Kunden die Möglichkeit zu bieten, die teuren Mainframe-Systeme an ihre Bedürfnisse anzupassen. Der Begriff ‚Software' wurde allenfalls als weitgefasstes Antonym zu Hardware benutzt – beispielsweise um nichttechnische Fehlerquellen wie "human factors in reliability" (Carhart 1953: 69) zu umschreiben. Erstmals im heutigen Sinne verwendet wurde der Ausdruck von dem Statistiker und Mathematiker John Tukey im Jahr 1958:

> "Today the 'software' comprising the carefully planned interpretive routines, compilers, and other aspects of automative programming are at least as important to the modern electronic calculator as its 'hardware' of tubes, transistors, wires, tapes, and the like." (Tukey 1958: 2)

Dass Software in dieser Anfangszeit der IT-Branche kein eigenständiger Marktwert zugesprochen wurde, liegt auch darin begründet, dass die einzelnen Computersysteme nicht kompatibel und deren situationsspezifische Einsatzpotenziale noch kaum erforscht waren, was nach einer intensiven Kooperation zwischen Anwendern und anbieterseitigen Spezialisten verlangte, um das jeweilige System überhaupt nutzbar zu machen (vgl. Phister 1976). Dieser hohe individuelle Programmieraufwand wurde von der *International Business Machines Corporation* (*IBM*) nach ihrem Einstieg in den Markt für digitale Computer 1952 als gravierendes Hemmnis für den Ausbau ihres Hardware-Geschäfts eingestuft, weshalb *IBM* mit *SHARE* bereits Mitte der 1950er-Jahre eine Gesellschaft für den Austausch von technischen Informationen zwischen Nutzern von *IBM*-Systemen mitbegründete. Ihre anfänglich zwei Dutzend Mitglieder entwickelten gemeinsam Modifikationen und Erweiterungen, allerdings wuchsen die Transaktionskosten in Relation zu den Produktivitätsgewinnen mit steigender Beteiligtenzahl disproportional an,

was mit ein Grund dafür war, dass das *SHARE*-Modell nicht zu einem Branchenstandard avancierte (vgl. Campbell-Kelly 2003; Brooks 1975).

Mit Unterstützung der *SHARE*-Gruppe entwickelte *IBM* die abstrakte Programmiersprache *FORTRAN*, die via Compilern erstmals die Möglichkeit bot, das gleiche Programm auf unterschiedlichen Systemen auszuführen, und gab den Programmcode mit Dokumentation ab 1957 kostenfrei an Kunden wie Mitbewerber ab, um – ähnlich wie in Beta-Programmen heute – praxisnahe Optimierungen zu ermöglichen und die eigene Marktposition zu stärken (vgl. Bashe et al. 1986). Systemübergreifende Sprachen wie *FORTRAN* senkten die Kosten in der Programmierung erheblich und machten zusammen mit den ersten Vereinheitlichungen im Hardwarebereich die Entwicklung standardisierter Applikationen durch Drittanbieter denkbar. Die erste kommerziell vertriebene Software war das Dokumentationssystem *AUTOFLOW* der Firma *Applied Data Research* zur automatischen Generierung von Programmablaufplänen, das ab 1965 für rund 2.400 US-Dollar angeboten wurde und 1968 das erste Patent für ein Computerprogramm erhielt. Viele *IBM*-Kunden konnten mit dem Konzept einer separat verkauften Anwendung jedoch zunächst wenig anfangen:

> "Over and over again, as a result of a sales presentation on AUTOFLOW, the potential customer would call his IBM account representative and ask when they were going to upgrade their program to do what AUTOFLOW did." (Johnson 2002: 107)

Bis in die 1960er-Jahre hinein wurde Software weder von Anbietern noch von Kunden als ein von der Hardware unabhängiges Produkt wahrgenommen, sondern "as a research tool to be developed and improved by all users" (Gulley/Lakhani 2010: 6). Dementsprechend lässt sich die frühe Softwareentwicklung als Spielart der durch Allen (1983) beschriebenen *collective invention* fassen: als zumeist informell strukturierte Kollaboration zwischen Produzenten und staatlichen bzw. universitären Einrichtungen, deren Kreis angesichts der erforderlichen finanziellen Ressourcen und Wissensbestände zunächst begrenzt blieb. Bereits in dieser Zeit nahm das Unternehmen *IBM* indes eine herausgehobene Stellung ein: 1960 erwirtschaftete *IBM* Einnahmen von ca. 1,8 Mrd. US-Dollar und produzierte 65 Prozent aller Computer auf dem US-amerikanischen Markt – ein Anteil, der sich bis 1965 auf knapp 75 Prozent steigern sollte (vgl. Mulligan 2013: 107).

2.2 Kommodifizierung — Subkulturgenese (1960er- und 1970er-Jahre)

Eine kommerzielle Softwareindustrie entwickelte sich zunächst primär in den USA (vgl. Mowery 1999). Da weder das Urheberrecht noch das Patentwesen auf die neue Produktkategorie eingestellt waren, blieb der Handel mit Software in den ersten Jahren freilich ein risikoreiches Geschäft, bis Ende der 1960er-Jahre erste Gerichtsurteile Rechtssicherheit schafften. Auftrieb erhielt die junge Branche nicht zuletzt durch ein 1969 eingeleitetes kartellrechtliches Verfahren, in welchem *IBM* vorgeworfen wurde, durch das kombinierte Angebot von Hardware, Software und Dienstleistungen potenzielle Mitbewerber aus dem Rennen werfen zu wollen. *IBM* kündigte daraufhin an, seine Angebote zu entbündeln, was sich zunächst weder lizenzarchitektonisch unkompliziert umsetzen noch problemlos an die Kunden vermitteln ließ (vgl. Burton 2002). Durch dieses und ähnliche Verfahren wurde Software zunehmend als eigenständiges Produkt sichtbar und rechtlich eingefasst. In den nachfolgenden Jahren erlebte die US-amerikanische Softwareindustrie ein rasantes Wachstum (s. Abb. 1).

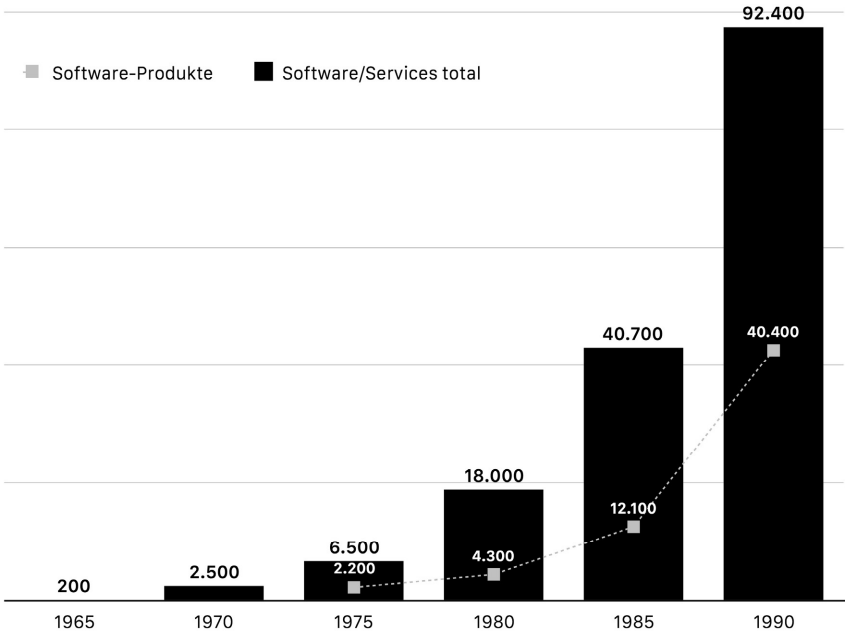

Abb. 1 Umsätze der US-amerikanischen Softwarebranche (in Mio. US-Dollar) Datenquelle: Computer & Business Equipment Manufacturers Association, Basis-USD-Werte für 1990 (vgl. Sayadian 1990; Juliussen/Juliussen 1990)

Während 1966 rund 50 unabhängige Unternehmen in der US-Softwarebranche tätig waren, stieg deren Zahl bis 1969 auf über 2.800 an, darunter als erfolgreichster Anbieter die *Computer Sciences Corporation*, die ihren Umsatz unter anderem mit Finanzverwaltungssoftware von 6 Mio. US-Dollar im Jahr 1964 auf 82 Mio. US-Dollar im Jahr 1970 steigern konnte (vgl. Fisher et al. 1983). Der Löwenanteil der Erlöse wurde aber schon damals in der Auftragsentwicklung und im Dienstleistungsgeschäft gemacht. Dass sich überhaupt ein nennenswerter Markt für Standardsoftware entwickeln konnte, lässt sich neben der primär durch *IBM* beförderten Vereinheitlichung der Mainframe-Systeme auf das Entstehen einer neuen Produktkategorie zurückführen: den schrankgroßen *minicomputer*. Das erste verkaufte Gerät dieses Typs war ab 1960 der durch die *Digital Equipment Corporation* (*DEC*) produzierte *PDP-1*, der 25.000 Operationen pro Sekunde ausführen konnte (0,025 MHz) und sich im Gegensatz zu früheren Computersystemen durch eine Einzelperson steuern ließ. Mitte der 1960er-Jahre waren Minicomputer für unter 20.000 US-Dollar zu haben; ihre Absatzzahlen überstiegen die der Mainframes Ende des Jahrzehnts (vgl. Steinmueller 1995).

Für die weitere Entwicklung der Softwarebranche und die Ausbildung einer computerzentrierten Subkultur spielte die Verbreitung von Minicomputern eine wichtige Rolle: Zum einen waren sie im Betrieb deutlich günstiger als Mainframes, daher nicht auf eine möglichst effiziente Nutzung ausgelegt und leichter zugänglich; zum anderen ermöglichten der *PDP-1* und seine Nachfolger die Verwendung neuer Ein- und Ausgabeschnittstellen, welche die Interaktion mit dem Nutzer vereinfachten und die Herausbildung neuer Softwaregenres (z. B. Grafikverarbeitung) beförderten. So wurden Fernschreiber für die Texteingabe und den Druck eingesetzt und Kathodenstrahlröhrenmonitore angeboten, die sich via ‚lightpen' bedienen ließen. Eines der bekanntesten *PDP-1*-Programme ist das Spiel *Spacewar!*, das ab 1961 durch Mitglieder des studentischen *Tech Model Railroad Clubs* am *MIT* für den von *DEC* an das Institut gespendeten Minicomputer entwickelt wurde und aufgrund seiner Demonstrationspotenziale rasch zur Standardsoftware des *PDP-1* gehörte (vgl. Lowood 2009).

Vor allem im nordamerikanischen akademischen Milieu, in dem sich *computer science* als akademische Disziplin bereits Ende der 1950er-Jahre etabliert hatte, während die Informatik in Deutschland erst rund zehn Jahre später entstand, boten stetig günstigere und institutsöffentlich erfahrbare Minicomputer einen idealen Nährboden für informelle Projektgruppen, deren Teilhabende sich ‚hackers' nannten und die Limitationen vorhandener Computer-

systeme zu überwinden suchten (vgl. Levy 1984). Auch die bis in die 1990er-Jahre populäre Programmiersprache *BASIC* wurde ab 1964 im Hochschulkontext entwickelt und kostenfrei abgegeben. Da *BASIC* rasch zu erlernen war, bildete es zusammen mit den ersten Mikrocomputern wie dem ab 1975 zehntausendfach verkauften *Altair 8800* (als Bausatz ab 400 US-Dollar) eine wesentliche Grundlage für die sich in den 1970er-Jahren entlang populärwissenschaftlicher Zeitschriften herausbildende Amateur-Computing-Szene:

> "It is these basement Edisons, part-time tinkerers and others who own computers for personal or professional reasons who will most probably realize the vast potential of the silicon chip for the consumer. They are an avid, eager-beaver breed, anxious to share technological insights and applications with other chip fanatics. Computerniks have already formed some 400 informal clubs, and these are growing rapidly." (Time Magazine 1978: 49)

Ein zentrales Problem, das für kommerzielle Softwareanbieter mit diesem neuen Markt einherging, bestand darin, dass Programme durch Computer-Hobbyisten zwar gerne untereinander weitergegeben, aber nur verhältnismäßig selten käuflich erworben wurden. Darüber beschwerte sich der junge Software-Entrepreneur Bill Gates 1976 mit Blick auf das durch sein 1975 gegründetes Unternehmen *Micro-Soft* verkaufte *Altair BASIC* in einen offenen Brief wie folgt:

> "As the majority of hobbyists must be aware, most of you steal your software. Hardware must be paid for, but software is something to share. Who cares if the people who worked on it get paid? Is this fair? [...] One thing you do is prevent good software from being written. Who can afford to do professional work for nothing? What hobbyist can put 3-man years into programming, finding all bugs, documenting his product and distribute for free? The fact is, no one besides us has invested a lot of money in hobby software. [...] Most directly, the thing you do is theft." (Gates 1976: 3)

Bill Gates' ‚Open Letter to Hobbyists' provozierte eine Vielzahl an Reaktionen, die entweder argumentierten, dass Mikrocomputer-Käufer schlicht ein funktionierendes System ohne Zusatzkosten erwarten könnten, *Altair BASIC* zu weiten Teilen auf Universitätsrechnern erstellt wurde und auf frei verfügbaren Vorarbeiten fuße oder dass es mithin an den Computernutzern selbst sei, kostenfreie Software zu entwickeln (vgl. Singer 1976; Warren 1976). Ein Beispiel für ein solches nutzerzentriertes Projekt aus dieser Zeit ist *Tiny BASIC*, welches durch Mitglieder der *Standford University* initiiert wurde und von einigen Autoren als ein Vorbote des ‚free software movements' angesehen wird (vgl. dazu kritisch: Driscoll 2015).

Ebenfalls eng mit dem universitären Milieu verknüpft war die Entwicklung des Betriebssystems *Unix*, das ab 1969 durch die *AT&T Bell Labs* entworfen wurde (vgl. Holtgrewe/Werle 2001; Weber 2005): 1973 öffentlich vorgestellt, stieß *Unix* auf ein breites Interesse, denn da es in der plattformübergreifenden Programmiersprache *C* verfasst war, konnte es auf nahezu allen Computern eingesetzt werden, wodurch Systeme unterschiedlicher Anbieter Kompatibilität erlangten und miteinander kommunizieren konnten. Weil es *AT&T* angesichts seiner Monopolstellung im Telefonmarkt bis in die 1980er-Jahre kartellrechtlich untersagt blieb, auch im IT-Bereich zu operieren, wurde *Unix* zunächst nicht kommerziell vertrieben, sondern mit Quellcode für einen symbolischen Betrag an Universitäten abgegeben, wo in der Folgezeit zentrale Systemerweiterungen wie etwa die Unterstützung der heute ubiquitär verwendeten TCP/IP-Protokollfamilie erarbeitet wurden. Die universitäre *Unix*-Entwicklung wurde durch die staatliche *(Defense) Advanced Research Projects Agency* unterstützt, die bereits seit 1958 intensiv in Softwareprojekte öffentlicher Einrichtungen investierte (darunter ab 1968 auch das *ARPANET*), deren Ergebnisse zumeist nach freien akademischen Prinzipien publiziert wurden (vgl. Mowery/Langlois 1996).

Zum einen wurde Software ab den 1960er-Jahren zunehmend als eigenständiges Produkt wahrgenommen; zum anderen bildeten sich erste computerzentrierte Subkulturen in geschützten Nischen heraus, die als Ausgangspunkt für spätere gefestigtere Gemeinschaften gelten können. Deren Beteiligte zeichneten sich durch eine hohe intrinsische Motivation aus, die Arbeitsgruppen waren im Regelfall informell strukturiert und die dort entwickelte Software stand meist zur freien Verfügung. Somit lassen sich diese Hobbyisten- und Hackergruppen in gewisser Hinsicht als frühe Varianten einer ‚peer production' in Benklers (2002) Sinne beschreiben. Neben Koordinationsproblemen bei steigender Projektgröße bestand aber nach wie vor die Gefahr der Proprietarisierung und Kommodifizierung ihrer Arbeitsergebnisse: *Unix* etwa wurde durch *AT&T* ab 1983 – sobald es kartellrechtlich möglich war – kommerzialisiert; der 1975 entstandene *Homebrew Computer Club* verlor seinen offenen Austauschcharakter, als mehr und mehr Teilnehmer eigene Firmen gründeten (darunter *Apple Computer Inc.*).

2.3 Institutionalisierung (1980er- und 1990er-Jahre)

In den 1980er-Jahren internationalisierte sich der Softwaremarkt zunehmend (s. Abb. 2) und die Hobbyistenkultur weitete sich nach Europa aus, was sich hierzulande unter anderem in der Gründung des *Chaos Computer Clubs*, dem Start der Fernsehsendung *WDR Computerclub* und der Einrichtung von „Computer-Centren“ (Spiegel 1983: 172) in Kaufhäusern niederschlug.

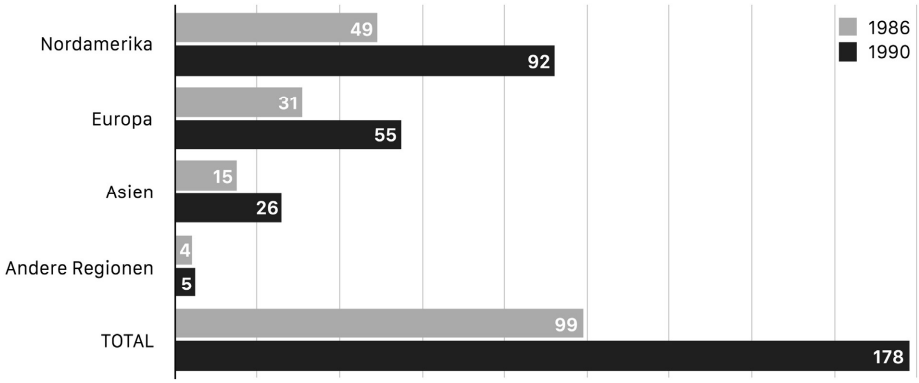

Abb. 2 Umsätze mit Software und Services weltweit (in Mrd. US-Dollar)
Eigene Berechnungen, Basis-USD-Werte für 1990. Datenquelle: Sayadian 1990

Als Treiber für die freie Softwareentwicklung erwiesen sich die ab 1980 in Expertenkreisen etablierenden neuartigen Formen elektronischer Vernetzung (darunter zuvorderst das *Usenet*), die den kostengünstigen Austausch von Daten über existente Telefonleitungen ermöglichten und so eine erste technische Lösung für die durch Frederick Brooks (1975) benannten Koordinationsprobleme in großen Projekten mit örtlich verteilten Entwicklern (vgl. Kap. 2.1) boten.

Das *Usenet* war auch die Plattform, über die der *MIT*-Mitarbeiter Richard M. Stallman 1983 sein *GNU*-Projekt (rekursives Akronym für ‚GNU’s not Unix‘) vorstellte, das als freies unixoides Betriebssystem eine Alternative zu proprietären Distributionen bieten wollte, deren Schutz und Ausschließbarkeit durch Gesetzesänderungen in den USA kurz zuvor noch einmal deutlich erhöht worden war (vgl. Menell 2002):

> “I consider that the golden rule requires that if I like a program I must share it with other people who like it. I cannot in good conscience sign a nondisclosure agreement or a software license agreement. So that I can continue to use computers without violating my principles, I have decided to put together a sufficient

> body of free software so that I will be able to get along without any software that is not free." (Stallman 1983)

Obgleich der angedachte komplett freie *GNU*-Kernel per se bis heute nicht für den praktischen Einsatz geeignet ist, wurden im Kontext des Projekts früh Setzungen vollzogen, welche die freie Softwareentwicklung nach wie vor prägen: 1985 gründete Stallman die *Free Software Foundation*, die seitdem das ‚movement' an freiwilligen Entwicklern juristisch und infrastrukturell unterstützt. Zu ihren ersten Großspendern gehörten 1988 *Sony* (10.000 US-Dollar, technisches Equipment) und 1989 *Hewlett-Packard* (100.000 US-Dollar), die als Hardwarehersteller an günstig lizenzierbarer Software interessiert waren (vgl. Schwarz/Takhteyev 2010; O'Mahony 2005). Zudem fungiert die *Free Software Foundation* als Interessenvertretung für freie Software und aktualisiert die ‚Free Software Definition', die in erster Version lautete:

> "The word 'free' [...] does not refer to price; it refers to freedom. First, the freedom to copy a program and redistribute it to your neighbors, so that they can use it as well as you. Second, the freedom to change a program, so that you can control it instead of it controlling you; for this, the source code must be made available to you." (Free Software Foundation 1986: 8)

Die bedeutsamste Neuerung, die durch die *Free Software Foundation* initiiert wurde, bestand jedoch – statt *GNU* schlicht als *public domain* zu deklarieren – in der Ausarbeitung rechtlich belastbarerer Lizenzen, die erzwingen, dass auch Weiterentwicklungen und Ableitungen freier Software stets frei bleiben müssen (‚Copyleft'). Nach der 1989 erstmals veröffentlichten *GNU General Public License* (*GPL*) dürfen entsprechend lizenzierte Programme modifiziert und weitergeben, aber keinesfalls mit eigenen Einschränkungen belegt werden:

> "Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein." (Free Software Foundation 1989)

Ab 2001 waren Verstöße gegen die *GPL* Gegenstand mehrerer Gerichtsverfahren in Europa und Nordamerika, in denen Unternehmen wie *Skype* oder *D-Link* gezwungen wurden, fragliche Softwareelemente in ihren Programmarchitekturen zu entfernen oder deren Quellcode wieder zur freien Verfügung zu stellen (vgl. Stiller 2011; Jaeger 2010). Osterloh/Rota (2007) und O'Mahony (2003: 1189) merken freilich an, dass für die Durchsetzung der in der

GPL angelegten Reziprozitätsprinzipien nicht nur die offizielle Gerichtsbarkeit, sondern ebenso "the court of public opinion" im *Usenet* bzw. später im *World Wide Web* eine tragende Rolle gespielt hat.

Der Erfolg des *GNU*-Projektes selbst blieb aufgrund seines Zuschnitts auf kostenintensive Workstations einerseits und seiner ideologischen Konnotationen andererseits begrenzt, da viele Entwickler auch für gut befundenen proprietären Code in ihre Architekturen integrieren wollten, was unter der *GPL* zunächst nicht möglich war (vgl. Weber 2000). Auf beide Problemstellungen bot das *Linux Kernel*-Projekt eine Antwort: *Linux* wurde 1991 durch den Studenten Linus Torvalds als freier Betriebssystemkern für die günstigeren *IBM*-kompatiblen Mikrocomputer vorgestellt und war daher für eine größere Zahl an Entwicklern attraktiv. Zudem zeichnete sich Torvalds von Anfang an durch eine liberalere Haltung als die *Free Software Foundation* aus, die sich in einem seiner Beiträge in der *Linux Kernel Mailinglist* wie folgt widerspiegelt:

> "I think ideology sucks. This world would be a much better place if people had less ideology, and a whole lot more 'I do this because it's fun and because others might find it useful, not because I got religion'." (Torvalds 2002)

Zwar wurde auch der *Linux Kernel* 1992 unter die *GPL* gestellt, da aber ein Kernel für ein lauffähiges Betriebssystem ohnehin stets mit weiteren Programmelementen zu einem Paket kombiniert werden muss, schloss dies die Verwendung proprietärer Komponenten zumindest nicht grundsätzlich aus. Oft wird der *Linux Kernel* mit freien *GNU*-Bibliotheken ergänzt; in vielen *Linux*-Distributionen (z. B. *Red Hat Enterprise Linux*) kommen aber auch herstellerspezifische Module zum Einsatz.

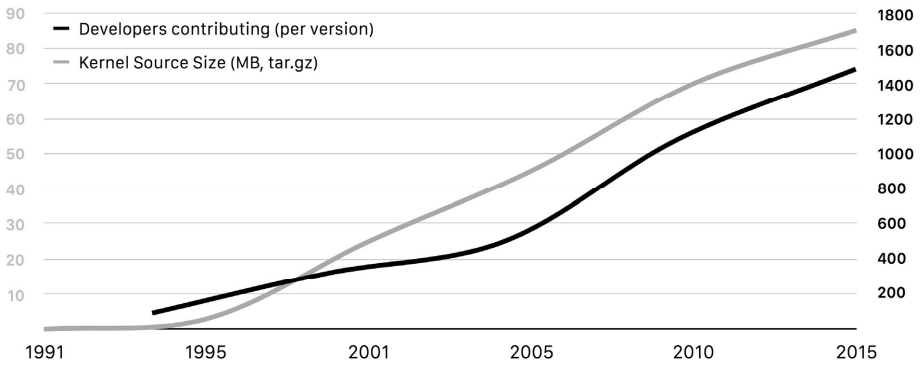

Abb. 3 Entwicklung der *Linux Kernel* Source Size. Datenquelle: Corbet/Kroah-Hartman/McPherson 2009–2015, https://kernel.org (Stand: 7/2015)

Ein Grund für das Florieren des *Linux*-Projekts (s. Abb. 3) bestand nach den eher überschaubaren Erfolgen anbieter- und plattformgebundener Onlinedienste in den 1980er-Jahren (z. B. *Bildschirmtext*, *AppleLink*, *Compuserve*) in der raschen Verbreitung des *World Wide Web*, das 1993 durch Tim Berners-Lee bzw. das *CERN* (*Conseil Européen pour la Recherche Nucléaire*) zur allgemeinen Nutzung freigegeben wurde. Sowohl der Zugriff auf *GNU/Linux* als auch die Beteiligung an entsprechenden Projekten und deren Koordination wurden durch die so möglich gewordenen neuen Austauschformen erheblich erleichtert. Überdies mussten sich freie Softwareprojekte für Mikrocomputer zu dieser Zeit in der Praxis nur noch auf eine Systemumgebung konzentrieren (s. Tab. 2): Während der globale Markt für Mikrocomputer 1984 überaus diversifiziert war (*Commodore*: 40 Prozent, *IBM*-PCs und Klone: 32 Prozent, *Apple*: 22 Prozent, *Atari*: 3 Prozent), lag der Anteil *IBM*-kompatibler Personal Computer 1995 bei über 90 Prozent (vgl. Reimer 2012). Nichtsdestotrotz blieb auch *Linux* zunächst ein lediglich in entsprechenden Expertenkreisen bekanntes Projekt.

Tabelle 2: Globale Marktanteile Microcomputer 1981–2011 (in Prozent)

	1981	1984	1988	1992	1996	2000	2004	2008	2011
IBM-kompatibel	3	32	79	87	95	97	98	97	96
Apple II	15	16	1	–	–	–	–	–	–
Apple Mac	–	6	6	11	5	3	2	3	4
Commodore 64 ff.	–	40	8	–	–	–	–	–	–
Commodore Amiga	–	–	3	2	–	–	–	–	–
Atari 400 ff.	21	3	–	–	–	–	–	–	–
Atari ST	–	–	2	1	–	–	–	–	–
andere Systeme	61	3	1	–	–	–	–	–	–

Quelle: Reimer 2012, eigene Recherchen

Dies änderte sich mit dem in der Computerszene wie auch in der allgemeinen Öffentlichkeit intensiv rezipierten Buch *The Cathedral and the Bazaar* (1999), das von dem Softwareentwickler Eric S. Raymond 1997 zunächst als Essay vorgestellt wurde. Raymond war von den Arbeitsweisen in Torvalds' Umfeld fasziniert:

> "What I saw around me was a community which had evolved the most effective software-development method ever *and didn't know it*! That is, an effective practice had evolved as a set of customs, transmitted by imitation and example,

> without the theory or language to explain why the practice worked." (Raymond 1998a: 96)

Seine Kernthese lautet: Während in klassischen Produktionsmodellen (‚cathedral') der Source-Code eines Programms allenfalls für fertige Versionen publiziert wird und die Entwicklergruppen hierarchisch organisiert sind, sei der Quellcode in Projekten wie *Linux* oder dem durch Raymond selbst initiierten *Fetchmail* stets einsehbar, ihre Gruppen seien horizontal strukturiert und geprägt durch modulare Selbstorganisation ohne zentrales Management (‚bazaar'). Kritische Beobachter stellten allerdings früh fest, dass in beiden Fällen zwar viele Vorschläge aus der Community kamen, die finalen Änderungen aber nur durch eine Person – Torvalds oder Raymond – freigegeben wurden (vgl. Connell 2000; Bezroukov 1999a). Anders formuliert: "The only entity that can really succeed in developing Linux is the entity that is trusted to do the right thing. And as it stands right now, I'm the only person/entity that has that degree of trust" (Torvalds 1998: 36).

Mit *GNU* und *Linux* entstanden in den 1980er- und 1990er-Jahren zwei Flaggschiffprojekte freier Softwareentwicklung, deren Erfolg durch die Effektivierung der Kommunikation im *Usenet* bzw. später im Web erheblich befördert wurde. In ihrem Kontext bildeten sich rechtliche Instrumente, welche die kollektiven Arbeitsresultate vor Proprietarisierung schützen, wie auch informelle Regeln heraus, deren Anerkennung sich im Onlinebereich unkomplizierter überprüfen ließ als zuvor. Daneben verfestigten sich erste anschlussfähige Narrative, die freie Softwareentwicklung als einen revolutionären, auf Peer-Review-Prozessen fußenden Produktionsmodus ohne Machtasymmetrien beschrieben und zeitweilig – trotz der aus dem akademischen Bereich bereits bekannten Limitationen solcher Verfahren (vgl. Bezroukov 1999b) – ohne weitere Rückfragen sozialwissenschaftlich weiterverarbeitet wurden (vgl. z. B. Benkler 2002, 2006).

2.4 Wachstum — Diversifizierung (2000er-Jahre)

Das Wachstum freier Softwareprojekte im nachfolgenden Jahrzehnt lässt sich neben der fortgesetzten Verbreitung des Internets vorrangig auf drei Dynamiken zurückführen:

(1) Zum ersten lagerte eine zunehmende Zahl an Firmen die Entwicklung von Softwareprodukten in den quelloffenen Bereich aus, darunter *Netscape Communications* als ein besonders früher und aufsehenerregender Fall: Nachdem es absehbar erschien, dass *Microsoft* den *Netscape Navigator* durch den in *Windows* integrierten *Internet Explorer* aus dem Markt drängen würde, kündigte das Unternehmen im Januar 1998 an, die Codebasis seines Browsers in das quelloffene Projekt *Mozilla* zu überführen, das bis 2003 durch *Netscape/AOL* über die *Mozilla Organization* unterstützt wurde, die danach in die *Mozilla Foundation* überging. Dabei ging es nicht zuletzt um die Erschließung neuer Kundenkreise: "By making our source code available to the Internet community, Netscape can expand its client software leadership by [...] building a community that addresses markets and needs we can't address on our own [...]" (Netscape 1998). Vier Jahre später wurde die *Mozilla Application Suite* veröffentlicht, aus der 2004 der Browser *Firefox* (bis 2003: *Phoenix*) ausgekoppelt wurde, der zu einem weiteren Vorzeigeprodukt freier Softwareentwicklung werden sollte.

(2) Angestoßen durch die *Netscape*-Entscheidung kam zum zweiten eine Gruppe um Eric Raymond Anfang 1998 zu dem Schluss, dass sich der politisch belegte Begriff ‚Free Software' für die weitere Verbreitung quelloffener Software in kommerziellen Kontexten als hinderlich erweisen könnte, schuf das neue Label ‚Open Source', das die Überlegenheit des Entwicklungsmodells betonen sowie gesellschaftsethische Aspekte ausblenden sollte (vgl. Raymond 1998b), und gründete mithilfe von Szeneprotagonisten wie Tim O'Reilly, der später auch den Begriff ‚Web 2.0' prägen sollte, die *Open Source Initiative* (*OSI*). Allerdings unterstützte die *Free Software Foundation* (*FSF*) diese Kursänderung nicht. Zwar unterscheiden sich die ‚Open Source Definition' der *OSI* (vgl. Perens 1999) und die ‚Free Software Definition' der *FSF* nur in wenigen Punkten; trotzdem grenzt Richard Stallman (2002: 57) ‚free software' deutlich von ‚open source' ab: "For the Open Source movement, non-free software is a suboptimal solution. For the Free Software movement, non-free software is a social problem and free software is the solution." Aus diesem Dissens entwickelte sich ein Ausrichtungsstreit, der heute mitunter durch Hybridakronyme wie *FOSS* (‚Free & Open Source Software') oder *FLOSS* (‚Free/Libre Open Source Software') umgangen wird.

(3) Zu den Vermarktungsbemühungen der *Open Source Initiative*, welche die unternehmerischen Vorteile quelloffener Software betonten, kamen zum dritten die durch den allgemeinen Dotcom-Boom beförderten Börsenerfolge einiger ‚open source companies' ab 1999 hinzu, darunter zuvorderst die

Linux-orientierten Hardwarehersteller *VA Linux* und *Cobalt Networks* sowie der Softwareanbieter *Red Hat*, der sich auf *Linux*-Architekturen für Unternehmen spezialisiert hat. Die Börsengänge dieser drei Firmen gehörten zu den erfolgreichsten Debüts aller Zeiten und erregten eine entsprechende mediale Aufmerksamkeit, die auf die Open-Source-Szene insgesamt abstrahlte (vgl. Gelsi 1999). Kurz danach beschrieb zum Beispiel der *Spiegel Linux* als „ernsthafte Konkurrenz zum Microsoft-Monopol“ (Spiegel 2000: 55; vgl. Spiegel 1999). Damit war ‚Open Source‘ als Schlagwort im öffentlichen Bewusstsein angekommen.

Diese ineinandergreifenden Dynamiken – die Überführung des *Netscape Navigators* in ein quelloffenes Projekt, die Etablierung des Labels ‚Open Source‘ und die anfänglichen Börsenerfolge *Linux*-orientierter Firmen – führten im Verbund mit der weiteren Ausweitung des Marktes (globale Ausgaben für Software/Services 2005: 885 Mrd. US-Dollar, 2010: 1.092 Mrd. US-Dollar; vgl. UNCTAD 2012) zu einem fast exponentiellen Wachstum freier Softwareprojekte sowohl mit Blick auf ihre Größe (s. Abb. 4) als auch auf ihre Anzahl: Während Ende der 1990er-Jahre einige hundert quelloffene Projekte existierten, waren 2008 auf der führenden Plattform *SourceForge* über 150.000 Projekte registriert. 2012 waren auf *GitHub* sowie *SourceForge* mehrere Millionen Repositorien und auf der Verzeichnisseite *Ohloh* (ab 2014: *Open Hub*) über 550.000 Projekte zu finden. Lediglich 47.000 dieser auf *Ohloh* katalogisierten Projekte wiesen jedoch Änderungen im vorangegangenen Jahr auf (vgl. Sands 2012). Der Verweis auf die Zahl registrierter Vorhaben per se sagt dementsprechend wenig aus.

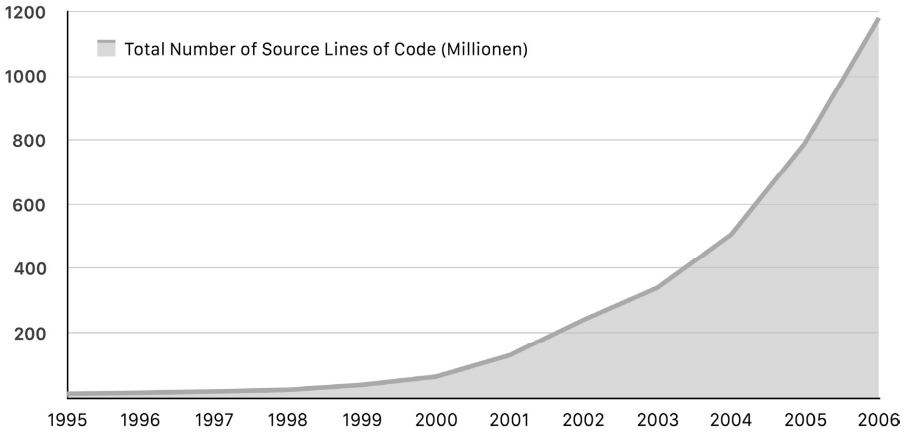

Abb. 4 Wachstum populärer Open-Source-Projekte (inkl. Copy & Paste)
Datenquelle: Deshpande/Riehle 2008 (5122 meistverlinkte Projekte auf *Ohloh*)

Vor dem Hintergrund der steigenden Zahl an Projekten, den Ausrichtungskonflikten in der Szene und der Definition eigener Lizenzmodelle durch Unternehmen und Stiftungen unterlag die quelloffene Softwareentwicklung in den 2000er-Jahren einer zunehmenden Diversifizierung, die sich in einem inzwischen sehr breiten Bouquet an freien Lizenzen niederschlägt (s. Tab. 3): Neben Copyleft-Lizenzen, die festlegen, dass auch Fortentwicklungen freier Software stets unter gleichen Bedingungen distribuiert werden müssen (*strongly protective*), sind Lizenzen getreten, welche die Einbindung freier Software in proprietäre Produkte erlauben, sofern ebendiese Komponenten quelloffen bleiben (*weakly protective*), oder wieder die Publikation von Weiterentwicklungen unter restriktiveren Bedingungen ermöglichen (*permissive*).

Tabelle 3: Meistgenutzte quelloffene Softwarelizenzen 2015 (2010)

	2015 (%)	2010 (%)	Ausrichtung	gebilligt durch	Publikation
GNU Public License 2.0	23	47	strongly protective	OSI, FSF	1991
MIT License (X11)	23	6	permissive	OSI, FSF	1988
Apache License 2.0	16	4	permissive	OSI, FSF	2004
GNU Public License 3.0	9	6	strongly protective	OSI, FSF	2007
BSD License 2.0 (3-clause)	6	6	permissive	OSI, FSF	1999
Artistic License 1 / [2.0]	5	9	[permissive]	OSI, [FSF]	2002
GNU Lesser GPL 2.1	4	9	weakly protective	OSI, FSF	1999
GNU Lesser GPL 3.0	2	< 1	weakly protective	OSI, FSF	2007
Microsoft Public License	2	2	permissive	OSI, FSF	2007
Eclipse Public License	2	< 1	permissive	OSI, FSF	2004
Code Project Open License	1	3	restrictive	–	2008
Mozilla Public License 1.1	< 1	< 1	weakly protective	OSI, FSF	1999

Quelle: *Black Duck* Knowledgebase; Webseiten der OSI und FSF (Stand: 10/2015)

Diese Vielfalt erweitert die strategischen Optionen insbesondere für kommerzielle Stakeholder (vgl. Lerner/Schankerman 2010; Lerner/Tirole

2005): Nachdem 2007 die *GPL* in dritter Version publiziert wurde und darin zuvor ausschöpfbare Lücken geschlossen worden waren, tauschte beispielsweise *Apple* die *GNU*-Compiler-Sammlung *GCC* in seiner Entwicklungsumgebung *Xcode* durch eine Lösung mit freizügiger Lizenz aus; *Google* entschied sich im Falle des mobilen Betriebssystems *Android* von vornherein dazu, den Großteil des Codes unter die permissive *Apache License 2.0* zu stellen.

Daneben lässt sich für die letzten 15 Jahre in zweierlei Hinsicht eine "corporatization of open source" (Hogan 2009) beobachten: Zum einen werden zentrale Projekte wie der *Linux Kernel*, *Mozilla Firefox*, der *Apache HTTP Server*, die Programmierumgebung *Eclipse* und die Cloud-Architektur *OpenStack* durch Spenden von Unternehmen mitfinanziert oder operieren wie die HTML Rendering Engine *WebKit* (*Apple*) und das mobile Betriebssystem *Android* (*Google*) unter der expliziten Federführung kommerzieller Anbieter (vgl. Fitzgerald 2006). Zum anderen speist sich die Entwicklerbasis großer Open-Source-Projekte zunehmend aus Unternehmenskontexten: Kolassa et al. (2014) kommen in ihren Analysen zum *Linux Kernel*-Projekt und 5000 weiteren auf *Open Hub* registrierten Vorhaben zu dem Schluss, dass zwischen 2000 und 2011 mehr als 50 Prozent aller Beiträge in der westlichen Kernarbeitszeit (Montag bis Freitag, 9 bis 17 Uhr, zeitzonenbereinigt) geleistet wurden; die *Linux Foundation* beobachtet, dass der Anteil unbezahlter, nicht unternehmensaffiliierter Beiträger an der Kernel-Entwicklung kontinuierlich abnimmt (2009: 18 Prozent; 2014: 12 Prozent), während der Anteil an Aktualisierungen von Entwicklern aus den mittlerweile rund 250 beteiligten Firmen weiter anwächst (vgl. Corbet/Kroah-Hartman/McPherson 2015).

In die *Linux Kernel*-Entwicklung involviert sind neben *Linux*-Distributoren wie *Red Hat* auch marktbestimmende IT-Konzerne. *IBM* (Umsatz 2014: 93 Mrd. US-Dollar) zeigt sich als einer der führenden Anbieter von Server-Hardware an der Entwicklung des *Linux Kernels* überaus interessiert und annoncierte zuletzt 2013 ein Investment über ca. 1 Mrd. US-Dollar in Open-Source-Technologien (vgl. IBM 2013). Aus ähnlichen Gründen investiert *Samsung* (Umsatz 2014: 188 Mrd. US-Dollar) in das Projekt: Die Firmware zahlreicher Produkte des Unternehmens fußt auf *Linux*; zudem hat *Samsung* 2011 zusammen mit *Intel* und weiteren Partnern die Entwicklung des *Linux*-basierten Betriebssystems *Tizen* angestoßen, um sich im Bereich der Mobile Devices und Medienabrufgeräte langfristig von *Android* absetzen zu können. Vergleichbare Interessen lassen sich für andere Hersteller identifizieren. Insgesamt sind Entwickler aus den vier Unternehmen *Intel*, *Red Hat*,

Samsung und *IBM* seit Version 3 für rund 25 Prozent der Änderungen im *Linux Kernel* verantwortlich (s. Tab. 4).

Tabelle 4: Änderungen im Linux Kernel nach Unternehmen/Organisationen

	2013–2014 (R 3.11–3.18)	**2011–2013** (R 3.0–3.10)	**2010–2012** (R 2.6.36–3.2)	**2005–2009** (R 2.6.11 bis 2.6.30)
unabhängig	12,4%	13,6%	16,2%	18,2%
nicht identifizierbar	4,9%	3,3%	4,3%	7,6%
Intel	10,5%	8,8%	7,2%	5,3%
Red Hat	8,4%	10,2%	10,7%	12,3%
Linaro	5,6%	4,1%	0,7%	n.a.
Samsung	4,4%	2,6%	1,7%	n.a.
IBM	3,2%	3,1%	3,7%	7,6%
SUSE/Novell	3,0%	3,5%	4,3%	7,6%
Texas Instruments	2,4%	4,1%	3,0%	n.a.
Vision Engraving	2,2%	2,3%	n.a.	n.a.
Google	2,1%	2,4%	1,5%	0,9%
Consultants	2,5%	1,7%	2,6%	2,5%
andere Organisationen	38,4%	40,3%	44,8%	38,0%
Intel, *Red Hat*, *Samsung* und *IBM* gesamt	26,5%	24,7%	23,3%	25,2%

Eigene Berechnungen. Datenquelle: Corbet/Kroah-Hartman/McPherson 2009–2015

In den letzten zwei Jahrzehnten konnte sich die quelloffene Entwicklung folglich als Methode zunehmend in der Softwarebranche etablieren. Sie hat dabei aber ihre Formatierung als Gegenentwurf zur proprietären Softwareentwicklung weitgehend verloren. Zwar lassen sich nach wie vor Liebhaberprojekte wie die vergleichsweise unregelmäßig aktualisierten *Linux*-Distributionen *Arch*, *Dragora* oder *Parabola* finden, die sich an den generischen Maximen freier Software ausrichten. In die meisten aktiven Open-Source-Projekte sind inzwischen jedoch etablierte Unternehmen involviert, die diese Kontexte nutzen, um außerhalb formaler Kooperationsbeziehungen mit externen Programmierern zu kollaborieren und so ihre ansonsten nach außen

eher abgeschotteten Forschungs- und Entwicklungsaktivitäten durch „kontrollierte Öffnungen an den Rändern“ (Dolata 2015: 17) zu erweitern. Im Unterschied zu den historischen Beispielen für ‚collective invention‘ (vgl. Tab. 1, S. 11) werden die Resultate dieser Kollaboration heute jedoch durch rechtlich belastbare Lizenzen geschützt, die sich zwar in unterschiedlichem Maße an den originären Reziprozitätsprinzipien freier Software ausrichten, aber stets die Möglichkeit zur Weitergabe und Modifikation einräumen.

Die konzeptionellen Ursprünge dieser Lizenzformen lassen sich in den frühen 1980er-Jahren verorten, als die Zeit, in der Software eher als ‚research tool‘ und weniger als ‚commodity‘ wahrgenommen wurde, noch nicht weit zurücklag und neue elektronische Vernetzungstechnologien zugleich erstmals die ortsunabhängige Koordination größerer Projektzusammenhänge ohne größeren eigenen infrastrukturellen Aufwand ermöglichten. In dieser Anfangsphase onlinebasierter freier Softwareentwicklung in einer von Marktmechanismen weithin abgelösten Nische lassen sich wohl am ehesten verbreitete Beispiele für eine von korporativen Interessen abgekoppelte ‚commons-based peer production‘ finden. Bereits ab Mitte der 1990er-Jahre wurden günstig lizenzierbare quelloffene Softwarekomponenten freilich zu Eckpfeilern einer internetzentrierten Start-up-Szene und in den Folgejahren stellen sich auch klassische Unternehmen auf die institutionellen Rahmenbedingungen freier Softwareentwicklung ein. Insofern beschreibt der Blogger Mike Bulajewski (2011) das Bild von Open-Source-Projekten als Gemeinschaften “of volunteer programmers collaborating together in a gift economy with no financial incentives” zurecht als Illusion. Vielmehr fußt die Vitalität vieler Projekte ganz wesentlich auf dem Engagement großer Konzerne, die ihre Ressourcen auf lange Sicht erwartungssicherer einbringen können als dies freiwilligen Einzelentwicklern in der Regel möglich ist.

3 Open Source und kommerzieller Markt

Quelloffene Softwareprojekte und kommerzieller IT-Markt sind heute eng ineinander verwoben (vgl. Westenholz 2012; Lerner/Schankerman 2010). Open-Source-Architekturen nehmen im Bereich der basalen informationstechnischen Infrastrukturen signifikante Marktanteile ein; auch und gerade in Unternehmen werden freie und proprietäre Programmelemente häufig in Kombination miteinander eingesetzt; kommerziell vertriebene Softwarepakete tragen oft Open-Source-Komponenten in sich *et vice versa*; Support- und Integrationsdienstleistungen um quelloffene Software sind – wenn auch weniger für dezidierte ‚open source companies' als für etablierte Anbieter – zu einem einträglichen Geschäftsfeld geworden. Überdies zeigt sich, dass die meisten marktrelevanten Open-Source-Projekte mittlerweile nicht mehr ohne das finanzielle wie personelle Engagement mittlerer und großer Unternehmen auskommen.

3.1 Marktrelevanz quelloffener Software

Der allgemeine Markt für Software und Services hat sich in den letzten Jahren erneut ausgeweitet (s. Abb. 5). Die weltweiten Erlöse mit ‚packaged software' bewegten sich 2014 je nach Schätzung zwischen 400 und 450 Mrd. US-Dollar (vgl. Gartner 2015; IDC 2015) und die größten softwarezentrierten Firmen machen inzwischen Jahresumsätze, die mit klassischen produzierenden Unternehmen wie *Bosch* (2014: 49 Mrd. Euro) oder *Airbus* (2014: 61 Mrd. Euro) vergleichbar sind (s. Tab. 5). Der Großteil der Umsatzes wird dabei im Enterprise-Bereich generiert: *Microsoft* (2015) setzte 2014 rund 19 Mrd. US-Dollar mit Consumer-Software und 42 Mrd. US-Dollar mit Unternehmenssoftware um; *IBM* (2015) generierte 2014 über 90 Prozent seines Softwareumsatzes mit Middleware und Betriebssystemen; *SAP*, *Oracle*, *EMC*, *Symantec*, *CA Technologies* und *Ericsson* operieren ebenfalls primär im Unternehmensbereich. Für dieses Segment diagnostizieren Marktforscher mittlerweile einen "widespread use of open-source technology in mission-critical IT portfolios" (Driver 2014) und sehen den Grund dafür nicht nur in Kostenvorteilen, sondern auch in der bedarfsbezogenen Anpassbarkeit quelloffener Architekturen (vgl. Aponovich et al. 2014).

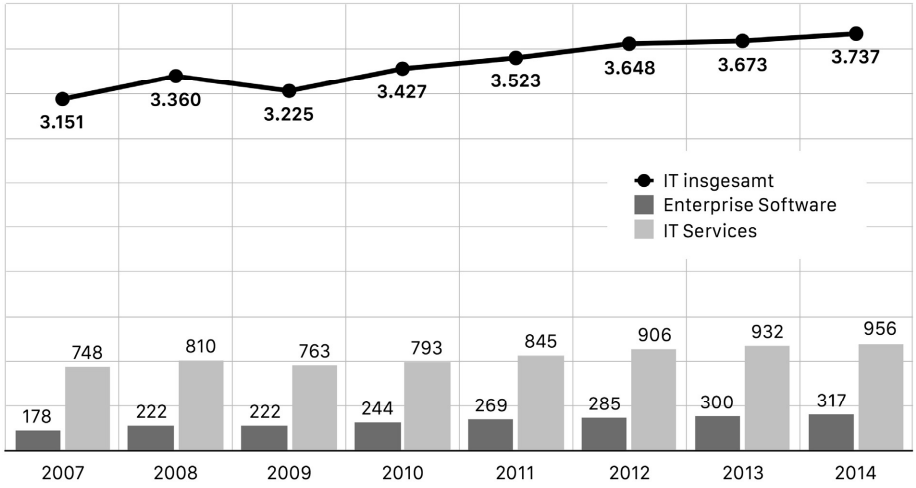

Abb. 5 Weltweite Ausgaben für IT in Mrd. US-Dollar 2007–2014
Quelle: Gartner Inc. Press Releases (Stand: 7/2015); IT gesamt: Hardware und Devices, Rechenzentren, Software, IT-Services, Telekommunikation

Tabelle 5:
Weltweit führende Softwarefirmen 2012 und 2014 (in Mrd. US-Dollar)

Unternehmen (Rang 2012)	Umsatz Software 2012*	Gesamt-umsatz 2012*	Umsatz-anteil Software 2012	Umsatz Software 2014**	Gesamt-umsatz 2014**	Aus-gaben für FuE 2014***
(1) *Microsoft*	58,4	72,9	**80 %**	60,8	86,8	11,4
(2) *IBM*	28,8	104,5	**28 %**	25,4	92,8	5,5
(3) *Oracle*	27,7	37,3	**74 %**	29,2	38,2	5,1
(4) *SAP*	16,6	21,3	**78 %**	15,9	18,9	2,5
(5) *EMC* (mit *VMware*)	9,3	21,7	**43 %**	n.a.	24,4	2,9
(6) *Ericsson*	8	34,9	**23 %**	6,3	26,2	4,2
(7) *Symantec*	6,4	6,8	**94 %**	n.a.	6,7	1
(8) *Hewlett-Packard*	5,5	119,2	**5 %**	3,9	111,5	3,4
(9) *Adobe Systems*	4,3	4,4	**98 %**	n.a.	4,1	0,8
(10) *CA Technologies*	4,3	4,6	**92 %**	4,1	4,5	0,6
(28) *Apple*	1,6	164,7	**1 %**	n.a.	182,8	6
(52) *Google*	0,8	50,1	**2 %**	n.a.	66	9,8
(74) *Amazon*	0,5	61,1	**1 %**	n.a.	89	9,2****

* PWC 2014; ** Form 10-K/Jahresberichte; *** FuE: Forschung und Entwicklung; **** “technology/content investment” (Amazon 2015: 19)

Ein Marktausblick der *International Data Corporation* (*IDC*), der sich mit der Frage beschäftigt, wie sich der globale Gesamtumsatz mit Anwendungs-, Administrations- und Systemsoftware nach adressierten Betriebsumgebungen aufteilt (s. Abb. 6), führt zwar zunächst die herausgehobene Stellung vor Augen, die *Microsoft Windows* als ‚operating environment' für Standardapplikationen 2014 mit 64 Prozent noch immer einnimmt. Und ebenso bleibt *Windows* im Bereich der Administrations- und Systemsoftware mit einem Anteil von über 50 Prozent laut dieser Studie die präferierte Einsatzumgebung, wobei 2016 immerhin 16 Prozent der Umsätze mit *Linux*-orientierten Programmen generiert werden sollen. Allerdings beziehen sich diese Projektionen ausschließlich auf marktlich gehandelte Standardsoftware und reflektieren weder das umsatzträchtige Feld der IT-Services noch den Einsatz kostenfreier Werkzeuge, Bibliotheken und Frameworks.

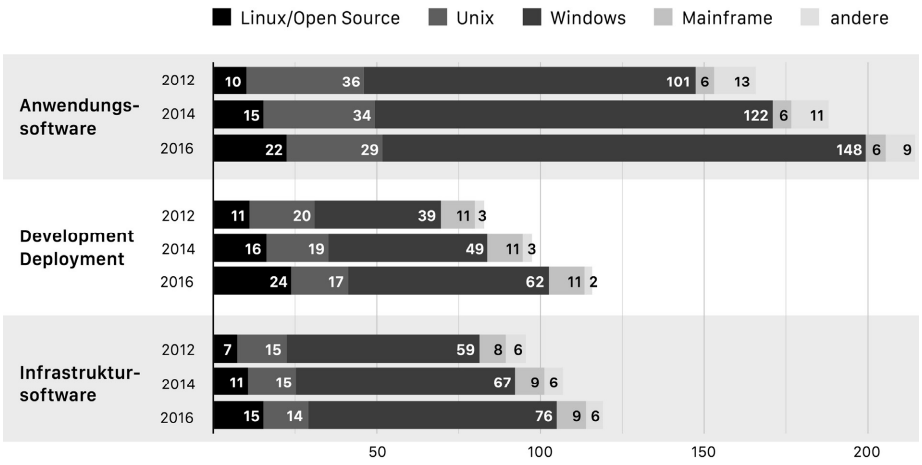

Abb. 6 Prognostizierter Umsatz nach Betriebsumgebungen (Mrd. US-Dollar)
Datenquelle: International Data Corporation 2012

Auf Tracking-Plattformen wie *StatCounter* oder *W3techs*, die öffentlich einsehbare Kenndaten von online-geschalteten Computern aggregieren, lässt sich indes nachvollziehen, dass Open-Source-Software inzwischen in vielen Marktsegmenten von signifikanter Bedeutung ist (s. Tab. 6). Dies gilt (abgesehen von *Android*) weniger für den Consumer-Bereich und Client-Computer (‚sichtbare Arbeitsrechner') als vielmehr für das Feld der serverseitigen Infrastrukturen: Während auf einem Personal Computer nach wie vor nur in Ausnahmefällen *GNU/Linux* läuft, *Microsoft Office* noch immer De-facto-Standard für Textverarbeitungs-, Tabellenkalkulations- und Präsentationspro-

gramme ist und sich der Marktanteil von *Mozillas Firefox*-Browser mit der Verbreitung von *Google Chrome* deutlich reduziert hat, nimmt *Linux* als Betriebssystem für öffentliche Server stabil zwei Drittel des Marktes ein; im Bereich des relationalen Datenbankmanagements hat sich der Anteil von Lösungen mit quelloffener Lizenz zuletzt auf knapp 45 Prozent erhöht; das Feld der HTTP-Server wird bereits seit den 1990er-Jahren von Open-Source-Architekturen beherrscht; und unter den verbreiteten Content-Management-Systemen hat *WordPress* seinen Vorsprung weiter ausgebaut.

Tabelle 6:
Geschätzte weltweite Marktanteile quelloffener Software (in Prozent)

	Open Source	2010	2015	Mitbewerber	2010	2015
Betriebssystem Personal Computer (a)	*GNU/Linux*	1	1,6	*MS Windows*	94	91
				Apple Mac OS X	5	7
Betriebssystem Mobile Devices (b)	*Android*	11	66	*Apple iOS*	30	19
				Symbian/ Nokia OS	33	3
				Blackberry	14	1
				Windows Phone	—	2
Webbrowser Desktop (c)	*Mozilla Firefox*	31	18	*Google Chrome*	14	57
				MS IE	47	17
				Apple Safari	5	4
Office Suites in Organisationen [2013] (d)	*Open Office / Libre Office*	2		*MS Office*	91	
				Google Docs	5	
Betriebssystem öffentliche Server (e)	*Linux* (inkl. unixoide Systeme)	69	67	*MS Windows*	31	33
Webserver [aktive Sites] (f)	*Apache*	72	56	*Microsoft IIS*	21	13
	Nginx	4	26	*Google Servers*	1	1
				LiteSpeed	1	2
Datenbank-management (g)	*OSS*-Lizenz (z. B. *MongoDB)*	35 (2012)	44	kommerzielle Lizenz (z. B. *Oracle*)	64 (2012)	56
Web-Content-Management-System (h)	*WordPress*	51	59	*Blogger (Google)*	2	3
	Joomla	12	7			
	Drupal	7	5	*Bitrix*	—	1
	Typo 3	4	2	*vBulletin*	8	1
				Shopify	—	1

Datenquellen (Stand: 9/2015): (a) NetApplications; (b, c) StatCounter; (d) Forrester 2013; (e, f, h) W3techs; (g) DB-Engines

Erhebungsdaten des Consultingunternehmens *Forrester* (2014) zeigen darüber hinaus auf, dass 2014 rund 80 Prozent der Softwareentwickler in Unternehmen weltweit auf Open-Source-Code zurückgegriffen haben – und zwar vorrangig im Datenbank-, Server- und Systemmanagement sowie in der Erstellung mobiler Anwendungen. Die gleiche Studie spricht quelloffenen Lösungen auf den Feldern des Cloud Computings (73 Prozent) und der Sicherheitsarchitekturen (59 Prozent) Marktführerschaft zu. Zudem kommt eine Erhebung unter Open-Source-Nutzern zu dem Ergebnis, dass universitäre und schulische Organisationen auf einigen Einsatzfeldern beinahe vollständig auf quelloffene Architekturen zurückgreifen (vgl. Black Duck/North Bridge 2015; ähnlich bereits: Ghosh et al. 2006). Dem Marktanalysten Jeffrey Hammond (2014: 21. Min.) zufolge geht mit dem korporativen Einsatz von Open-Source-Software freilich nicht nur die Hoffnung auf eine signifikante Kostenreduktion, sondern auch auf einen deutlich vereinfachten Anschaffungsprozess einher:

> "[Developers] are basically saying: Hey, we're trying all the open source alternatives first, because it's so painful dealing with our procurement department, that we'll only go commercial if we can prove that the open source stuff can't work." (Hammond 2014: 21. Min.)

Diesen Spielraum haben Entwickler, da viele Organisationen bislang keine dezidierten Richtlinien für den Einsatz buchhalterisch zunächst kostenfreier Open-Source-Software ausgearbeitet haben. Aus der bloßen Entscheidung für quelloffene Software allerdings unmittelbar auf benennbare Einsparungen zu schließen (so z. B. Walli et al. 2005; Riehle 2013), wirkt aus praktischer Sicht verfehlt: Auch wenn sich der Programmcode selbst unentgeltlich nutzen lässt, schmälert dies nicht zwangsläufig die technischen Integrations- und personellen Trainingskosten; ferner kann sich mitunter der Betrieb gegenüber proprietären Lösungen verteuern, da Open-Source-Produkte meist keiner expliziten Gewährleistungs- und Supportpflicht unterliegen.

Zusammengenommen sprechen die reflektierten Daten und Statistiken für die u. a. auch durch die Marktforschungsunternehmen *Gartner* und *Forrester* vertretende Einschätzung, dass quelloffenen Softwarearchitekturen inzwischen vor allen Dingen im Bereich der grundlegenden kommunikations- und informationstechnischen Infrastrukturen eine hohe Marktrelevanz zugesprochen werden kann und es daher insbesondere für größere privatwirtschaftliche sowie staatliche Organisationen zunehmend unmöglich wird, nicht auf Open-Source-Architekturen zurückzugreifen: "By 2016, at least 95 percent of IT organizations will leverage nontrivial elements of open-source software

technology [...] including cases where they might not be aware of it" (Driver 2013: 2. Min.). Vor diesem Hintergrund kommen auch schon lange tätige Hersteller proprietärer Softwareprodukte wie *Microsoft*, *IBM*, *Adobe*, *Oracle* oder *SAP* nicht umhin, sich auf dieses veränderte Umfeld einstellen.

3.2 Involvement etablierter Anbieter

Dementsprechend sind die meisten großen Softwareanbieter heute intensiv in quelloffene Projekte involviert. *Microsoft* – das Unternehmen, das Open Source lange als "intellectual-property destroyer" (Leonard 2001) bezeichnet hat und nach wie vor die "competitive advantages" (Microsoft 2015a: 12) hervorhebt, die aus geschützter interner Entwicklungsarbeit resultieren – betreibt mit *CodePlex* seit 2006 eine Web-Plattform für Open-Source-Vorhaben, hat 2012 die Tochterfirma *MS Open Technologies* lanciert, gehörte zwischenzeitig zu den Top-Beiträgern in der *Linux Kernel*-Entwicklung (vgl. Corbet/Kroah-Hartman/McPherson 2012) und hat in den letzten Jahren das Software-Framework *.NET* sowie zahlreiche weitere Komponenten unter quelloffene Lizenz gestellt. Dabei geht es *Microsoft* vor allem anderen darum, angesichts zunehmend heterogener IT-Infrastrukturen im Enterprise Bereich übergreifende Standards abzusichern sowie die Kompatibilität und Interoperabilität eigener Produkte zu erhöhen:

> "At times, we make select intellectual property broadly available at no or low cost to achieve a strategic objective, such as promoting industry standards, advancing interoperability, or attracting and enabling our external development community." (Microsoft 2015a: 13)

Microsofts Linux-Engagement etwa lässt sich primär auf die Integration von Treibern für die Virtualisierungstechnik *Hyper-V* als Bestandteil von *Windows Server* zurückführen (vgl. McAllister 2013); zusammen mit *Google*, *IBM* und anderen Unternehmen hat *Microsoft* 2014 unter dem Eindruck des schwerwiegenden *OpenSSL*-Fehlers ‚Heartbleed' darüber hinaus die *Core Infrastructure Initiative* (2015) geformt, um "open source projects that are in the critical path for core computing functions" langfristig zu fördern und auf diese Weise deren Verlässlichkeit abzusichern.

Welche Anteile ihrer Ausgaben für Forschung und Entwicklung *Microsoft* und andere marktführende Unternehmen in Open-Source-Projekte investie-

ren, lässt sich aus den jährlichen Geschäftsberichten nicht herauslesen und letztlich kaum gesondert abschätzen, da quelloffene Komponenten heute für viele herstellerspezifische Softwareprodukte eine tragende Rolle spielen. *Apples* Betriebssystempakete liefern dafür ein anschauliches Beispiel: Sowohl *Mac OS X* als auch *iOS*, *watchOS* und *tvOS* basieren auf dem freien unixoiden Betriebssystemkern *Darwin* und tragen mehr als 200 weitere Open-Source-Komponenten mit sich, so zum Beispiel die HTML Rendering Engine *WebKit*, die in vielen kommerziellen Produkten Anwendung findet (z. B. von *Sony*, *Samsung*, *Google*, *Amazon*). Korrespondierend dazu speist sich die Entwicklerbasis von *Darwin* fast ausschließlich aus dem Unternehmen *Apple*; der weit überwiegende Teil der Änderungen im *WebKit*-Projekt zwischen 2002 und 2013 stammt von bei *Google* oder *Apple* angestellten Mitarbeitern (vgl. Bitergia 2013).

Auch das Unternehmen *IBM*, das seit jeher den Großteil seines Umsatzes mit Hardware- und Softwarelösungen für Geschäftskunden verdient, beteiligt sich intensiv an Open-Source-Projekten: Bereits zur Jahrtausendwende investierte *IBM* mehrere 100 Mio. US-Dollar in *Linux*-Entwicklungsprogramme und portierte seine Software für *GNU/Linux* (vgl. Capek et al. 2005). Zum ersten verfolgte *IBM* damit das Ziel, *Microsofts* Dominanz im Bereich der Enterprise-Software entgegenzusteuern:

> "The arrival of Linux and the open source movement was well-timed for IBM [...]. Here is a software development and distribution model that potentially threatens the core of the Microsoft business model. Of course, it also threatens the business models of Microsoft's competitors, but one thing at a time. So let's embrace it, win over a new generation of developers and ultimately computer users, slow down the pace of Windows 2000 acceptance in the enterprise, and live to fight another day." (Desmond 2001: 4)

Zum zweiten ging es *IBM* wie schon im Falle von *FORTRAN* (vgl. Kap. 2.1) darum, das Servicegeschäft um proprietäre Technologien im Verbund mit quelloffener Software auszubauen und die Verbreitung *IBM*-naher Standards zu erhöhen. Und zum dritten konnte *IBM* als Arbeitgeber in Entwicklerkreisen früh von dem Image "[of] being a patron for industry-wide open source efforts" (Coté et al. 2007: 4; vgl. Fitzgerald 2006) profitieren.

Inzwischen ist *IBM* in über 100 Open-Source-Projekte involviert, darunter auch die seit 2010 entwickelte Cloud-Computing-Architektur *OpenStack*, welche seit 2013 die Grundlage für sämtliche *IBM Cloud Services* bietet. "IBM clearly wants to influence OpenStack's technological direction and efforts to develop industry standards for cloud computing, which is still a

relatively immature architecture" (Gonsalves 2013). Neben *IBM* sind auch die multinationalen IT-Konzerne *Intel* und *Hewlett-Packard* intensiv an der Entwicklung von *OpenStack* beteiligt (s. Abb. 7). Ihr Engagement resultiert jedoch ebenfalls nicht aus Idealismus, sondern aus unternehmerischem Kalkül: "Such actions are comparable to giving away the razor (the code) to sell more razor blades (the related consulting services that IBM and HP hope to provide)" (Lerner 2012: 43). Im Falle von *OpenStack* erleichtert die permissive Lizenzierung des Projektes zudem die Überführung in herstellerspezifische Lösungen wie die Enterprise-Cloud-Plattform *HP Helion*.

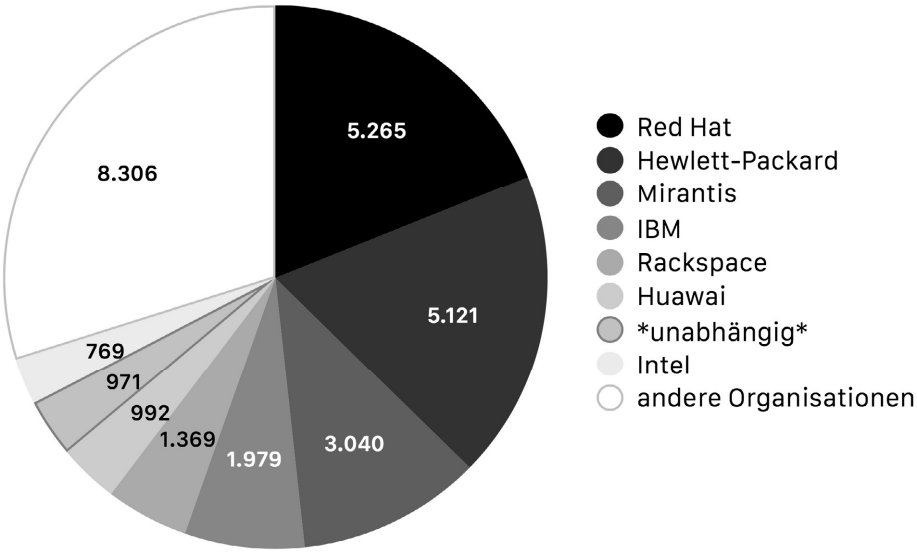

Abb. 7 Commits in *OpenStack* – Release Liberty (2015)
Datenquelle: http://stackalytics.com

Aus vergleichbaren Gründen beteiligen sich andere Firmen an Open-Source-Projekten: *Oracle* verdient sein Geld seit den 1970er-Jahren mit Datenbanklösungen für staatliche und privatwirtschaftliche Einrichtungen, ist in diesem Kontext auf die Interoperabilität seiner Komponenten mit quelloffenen Infrastrukturen angewiesen und verfolgt mit dual lizenzierten Produkten wie *MySQL* überdies das Geschäftsmodell, erweiterte Enterprise-Versionen von Open-Source-Software inklusive Supportdienstleistungen zu verkaufen. Auch *SAP* ist als führender Anbieter von Enterprise-Resource-Planning-Systemen darauf bedacht, die Entwicklung quelloffener Software zu beobachten und mitzugestalten, um die Kompatibilität eigener Produkte abzusichern sowie „Kunden aufzuzeigen, dass sie ihr Open-Source-Know-how

auch in SAP-Umgebungen sinnvoll einsetzen können“ (Spies 2010). *Adobe* hingegen sieht sein Open-Source-Involvement vorrangig “as part of a strategy to assist and grow the developer community in line with the company’s products and markets” (Germain 2014).

Eine spezielle Variante korporativen Open-Source-Engagements stellt die Entwicklung des *Linux*-basierten Betriebssystems *Android* durch die *Open Handset Alliance* seit 2007 dar: Beworben als “open-source software stack [...] to make sure there was no central point of failure, where one industry player could restrict or control the innovations of any other” (http://source.android.com) und in der Literatur oft in eine Linie mit dem *Linux Kernel* oder dem *Apache HTTP Server* gestellt (vgl. Herstatt/Ehls 2015: XVII), wird das Projekt de facto alleinig von *Google* gesteuert:

> “Google largely follows the ‘cathedral’ model of software production; only the source code of completed versions of the OS is released. Because it fully controls the development of the OS, Google can determine the technological specifications to which Android partners must abide.” (Spreeuwenberg/Poell 2012)

Mit der Initiierung des *Android*-Projekts ging es *Google* (2015a) mit offenkundigem Erfolg vor allen Dingen darum, den nahtlosen Zugriff auf eigene Dienste und Plattformen wie *Google Play* auf möglichst vielen Geräten zu ermöglichen: Während *Google* 2007 knapp 99 Prozent seines Umsatzes (16,6 Mrd. US-Dollar) mit Werbung generierte, war der Verkauf digitaler Inhalte 2014 für rund 11 Prozent des Jahresumsatzes (66 Mrd. US-Dollar) verantwortlich.

Davon abgesehen lässt sich das Involvement großer Softwarekonzerne in die quelloffene Entwicklung zum einen aus der Notwendigkeit ableiten, die Position eigener Lösungen in einem zunehmend von Open-Source-Komponenten durchdrungenen IT-Umfeld abzusichern. Zum anderen ist es traditionell primär im Enterprise-Bereich tätigen Anbietern wie *HP*, *IBM* und *Oracle* in den letzten Jahren gelungen, ein Dienstleistungsgeschäft um Open-Source-Architekturen und deren Integration mit proprietären Komponenten zu etablieren. Vor allen Dingen für kleinere Softwareunternehmen dient das Engagement in Open-Source-Projekten oder die Gründung eigener Vorhaben darüber hinaus als “marketing tool to increase brand recognition” (Dahlander/Magnusson 2008: 638): Gelingt es, das eigene Unternehmen mit prominenten Projekten zu assoziieren und davon ausgehend Fachbücher oder Konferenzbeiträge zu lancieren, kann dies gegenüber Kunden und externen Entwicklern erheblich zur Steigerung des Firmen-Renommees beitragen.

3.3 Open-Source-Unternehmen

Neben den genannten hybriden Verwertungsmodellen etablierter Anbieter haben sich Ende der 1990er-Jahre eine Reihe reiner ‚open source companies' herausgebildet, die ihr Kernprodukt – den Softwarecode – kostenfrei abgeben und versuchen, mit auf den Unternehmenseinsatz zugeschnittenen Editionen im Verbund mit Supportdienstleistungen ein Geschäft aufzubauen. Mit Ausnahme von *Red Hat* sind die meisten dieser im Fahrwasser des New-Economy-Hypes gegründeten Open-Source-Firmen allerdings rasch wieder eingegangen oder von größeren Unternehmen übernommen worden: *Sleepycat* (Datenbankmanagement) etwa gehört (wie *MySQL AB*) seit 2006 zu *Oracle*; *JBoss* (Middleware) wurde durch *Red Hat* aufgekauft; *SpringSource* (*Java*-Framework) ist seit 2008 Teil von *VMware*. Pointiert zusammengefasst:

> "[...] beyond Red Hat the effort [of commercializing open source] has largely been a failure from a business standpoint. Consider that the 'support' model has been around for 20 years, and other than Red Hat there are no other public standalone companies that have been able to offer an alternative to their proprietary counterpart." (Levine 2014)

Red Hat bietet inzwischen ein breites Portfolio an unternehmenszentrierter Software inklusive Support auf Subskriptionsbasis an – von Cloud Computing Architekturen bis hin zu Lösungen für die Administration und Entwicklung. Den Großteil seines jährlichen Umsatzes (2014 geschätzte 87 Prozent) generiert das Unternehmen jedoch mit seinem Betriebssystem *Red Hat Enterprise Linux*, das durch integrierte Systemmanagement-Dienste die Verwaltung von IT-Netzwerken in Großorganisationen erleichtert. Zu seinen prominentesten Kunden zählen das *US Department of Defense*, das Filmstudio *DreamWorks* und die New Yorker Börse; ferner unterhält *Red Hat* Partnerschaften mit *Amazon*, *HP*, *Cisco*, *IBM* und weiteren führenden Unternehmen. Nach Schätzungen der *IDC* hielt *Red Hat* 2014 einen Anteil von 64 Prozent auf dem weltweiten Markt für kommerzielle *Linux*-Distributionen (vgl. Ante 2014).

Der in dieser Form singuläre Erfolg von *Red Hat* (s. Abb. 8) kann auf die generell zunehmende Adaption von *Linux* im Enterprise-Bereich zurückgeführt werden, die sich nicht zuletzt mit der erhofften Vermeidung von ‚vendor lock-ins' und Einsparungspotenzialen begründen lässt, aber erwartungssichere Supportleistungen keineswegs obsolet macht. In diesem Kontext konnte *Red Hat* gegenüber später hinzugekommenen Mitbewerbern einen

First-Mover-Vorteil ausspielen, da *Red Hat Linux* und die Verwaltungsplattform *Red Hat Network* bereits ab 2000 ein stetig weiterentwickeltes Gesamtsystem bildeten und das Unternehmen frühzeitig Kooperationen mit marktdominanten Hard- und Software-Anbietern eingegangen ist (vgl. West/Dedrick 2001). Neben *Red Hat* und kleineren Anbietern wie *SUSE* (Umsatz 2013: 200 Mio. US-Dollar) und *Canonical* (Umsatz 2013: 66 Mio. US-Dollar) wird der Weltmarkt für *Linux* (Hardware, Software, Services), dessen Gesamtumsatz für 2016 auf 70 Mrd. US-Dollar (2012: 35 Mrd. US-Dollar) geschätzt wird (vgl. Gillen 2013), durch die entsprechenden Divisionen größerer Konzerne bestritten, die intensiver in Forschung und Entwicklung bzw. die Vermarktung ihrer Produkte investieren können, als dies kleineren Unternehmen wie *Canonical* oder auch *Red Hat* möglich ist.

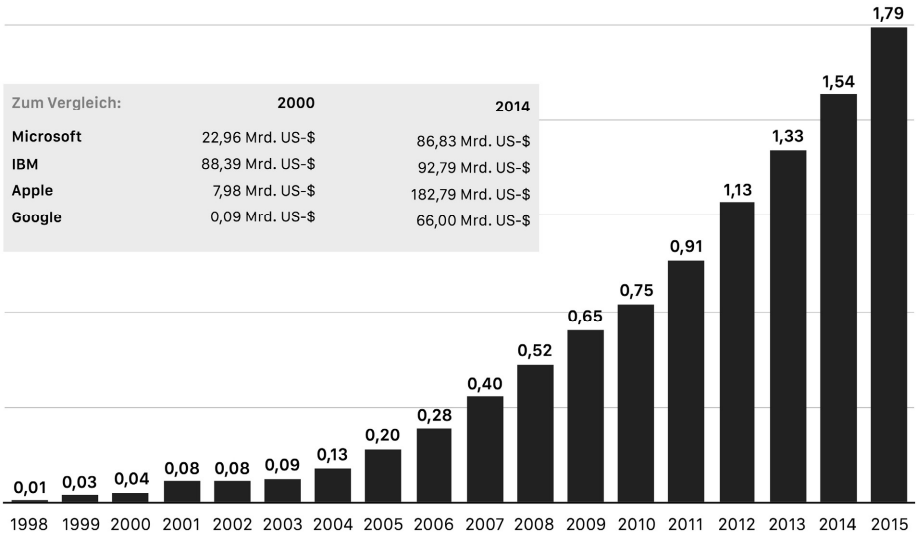

Abb. 8 Red Hat-Jahresumsätze (Mrd. US-Dollar; fiscal years ending 28. Feb.)
Datenquelle: Jahresberichte der Unternehmen

Außerhalb der *Linux*-Entwicklung sind im Open-Source-Umfeld in den letzten zehn Jahren eine Vielzahl neuer Start-up-Firmen entstanden, die zwar mitunter beträchtliche Investorengelder einwerben konnten, aber oft nicht profitabel operieren. Das Unternehmen *Hortonworks* (2015) beispielsweise, das auf der Basis von *Apache Hadoop* Lösungen zur distribuierten Verarbeitung großer Datenmengen anbietet und zwischen 2011 und 2014 rund 250 Mio. US-Dollar an Investorengeldern erhalten hat, wies für 2014 einen Umsatz von 46 Mio. US-Dollar (2013: 24 Mio. US- Dollar) bei Betriebskosten

von 135 Mio. US-Dollar (2013: 68 Mio. US-Dollar) aus; die seit 2011 an der Börse notierte Firma *Jive Software* (2014), die auf Wissensmanagement-architekturen für den geschäftlichen Bereich spezialisiert ist, machte 2014 einen Verlust von 55 Mio. US-Dollar (2013: 75 Mio. US-Dollar). Daneben existieren eine Reihe privat gehaltener open-source-affiner Start-ups, die ihre Kennzahlen nicht öffentlich ausweisen, aber ebenfalls zumeist (noch) nicht selbsttragend sind, darunter etwa *SugarCRM* (Kundenmanagement) oder *Alfresco* (Dokumentenverwaltung).

In ihrer Außendarstellung verzichten viele dieser Firmen mittlerweile allerdings auf ‚Open Source' als primäres Differenzierungsmerkmal zugunsten klassischer Nutzenversprechen (z. B. Effizienz, Reliabilität, Kostenvorteile). Wie Magnus Bergquist und Kollegen (2012) in qualitativen Interviews mit Softwareentwicklern in ‚open source companies' festgestellt haben, zeichnen sich solche Unternehmen oftmals durch eine nur noch geringe Loyalität gegenüber den originären (und in den entsprechenden Lizenzmodellen angelegten) Reziprozitätsidealen freier Software aus:

> "There is a tension between the GPL [GNU General Public License] and business which has consequences for what we can do and what we want to do. At the end of the day the company must earn money to survive. Richard Stallman has a very idealistic view of the world, which is admirable. But if one considers it from a business perspective one realizes that it is not feasible in practice." (Open Source Service Platform Provider, in Bergquist et al. 2012: 8)

Stattdessen sind es heute vor allem große Anbieter wie *IBM* ("Open Source & Standards are key to making our planet smarter") oder *Microsoft* ("Openness builds bridges between platforms and people"), die in ihrer Öffentlichkeitsarbeit auf ausgewählte Maximen freier Softwareentwicklung verweisen (vgl. IBM 2015b; Microsoft 2015b).

3.4 Patronage und Spendenfinanzierung

Imagepflege ist allerdings nur einer der Gründe, warum führende IT-Konzerne neben ihrem Involvement in die aktive Code-Entwicklung als Sponsoren und Partner für ein breites Portfolio an Open-Source-Vorhaben auftreten. In vielen Fällen eröffnet ein finanzielles Engagement den investierenden Unternehmen überdies die Möglichkeit, die Ausrichtung der entsprechenden Projekte im- oder explizit im Sinne ihrer Partikularinteressen mitzugestalten

(etwa durch einen Sitz im Managing Board). Eine exemplarische Zusammenschau der Open-Source-Projekte, die unter den Nutzern der Katalogisierungs-Website *Open Hub* in der ersten Jahreshälfte 2015 am populärsten waren und zwischen 5/2014 und 5/2015 mindestens 2.000 Änderungen aufgewiesen haben (s. Tab. 7), führt in diesem Zusammenhang vor Augen, dass der überwiegende Teil der dort aufgelisteten marktrelevanten und regelmäßig aktualisierten Projektgemeinschaften nicht ohne die finanzielle Unterstützung großer IT-Unternehmen auskommt, darunter in vielen Fällen *Google*, *IBM* und *Hewlett-Packard*.

Tabelle 7: Populäre Open-Source-Projekte auf Open Hub

Projekt	Commits (5/2014 bis 5/2015)	assoziierte Organisation	Finanzierung (neben Entwicklungsarbeit)
Android (mobiles OS)	81.119	*Google*, *Open Handset Alliance* (Konsortium)	
KVM (Linux Virtual Machine) **Linux Kernel**	70.730 65.402	*Linux Foundation*	Spenden, Mitgliederbeiträge (u. a. *HP*, *Intel*, *IBM*, *NEC*, *Oracle*, *Samsung*, *Qualcomm*)
OpenStack (Cloud Computing)	62.370	*OpenStack Foundation*	Mitgliedererbeiträge (u. a. *HP*, *IBM*, *Intel*, *Red Hat*)
Chromium (Browser)	61.454	*Google*	*Google* et al.
Mozilla Core (Bibliothek) **Mozilla Firefox** (Browser)	60.391 60.226	*Mozilla Foundation*	Provisionen (2013: 95% *Google*), Spenden, Einblenden von Werbung
GNOME (Unix/Linux Desktop)	40.853	*GNOME Foundation*	Spenden (u. a. *Google*, *IBM*, *Intel*, *FSF*)
KDE (Arbeitsplatzumgebung)	36.220	*KDE e.V.*	Patronagen (u. a. *Google*, *SUSE*)
Debian (GNU/Linux OS)	26.374	*Debian Project*	Spenden, Partner (u. a. *HP*, *1&1*)
LibreOffice (Bürosoftware)	21.384	*Document Foundation*	Spenden (u. a. *Google*, *Red Hat*, *Intel*)
WebKit (HTML Rendering)	14.867	*Apple*, *Google*, *Amazon*, *Sony* u. a.	
PHP (Skriptsprache)	9.385	*PHP Group*	Spenden, technische Ressourcen (u. a. *Facebook*)

Forts. der Tab. auf S. 44 →

Projekt	Commits (5/2014 bis 5/2015)	assoziierte Organisation	Finanzierung (neben Entwicklungsarbeit)
GNU Compiler Collection	8.382	*Free Software Foundation*	Patronagen (u. a. *Google*, *IBM*), Spenden
Python (Programmiersprache)	5.208	*Python Foundation*	Sponsoring (u. a. *Google*, *HP*, *Microsoft*)
MySQL (relationale Datenbank)	4.917	*Oracle Corporation*	
X.Org (Interface-Protokoll)	4.487	*X.Org Foundation*	Spenden, Mitgliederbeiträge (u. a. *HP*, *IBM*, *AMD*)
WordPress (CMS)	3.621	*WP Foundation*	*Automattic Inc.*, Veranstaltungen, Spenden
VLC Media Player	3.043	*Video Lan NPO*	Nutzerspenden, Partnerprogramme
Eclipse (IDE)	3.515	*Eclipse Foundation*	Mitgliederbeiträge (u. a. *Google*, *IBM*, *Oracle*, *SAP*)
Apache HTTP Server **Apache Tomcat**	2.356 3.721	*Apache Foundation*	Sponsoring (u. a. *Google*, *Microsoft*, *Facebook*, *Yahoo!*), Veranstaltungen
OpenSSL (Sicherheitsprotokoll)	1.467	*OpenSSL Team*	seit 2014: Core Infrastructure Initiative, Spenden
Arch Linux (OS)	1.323	—	Einzelspenden

Quelle: *Open Hub*, Jahresberichte (Stand: 7/2015; ohne Frameworks, Distributions)

Korporativ initiierte Entwicklungsprojekte stehen naturgemäß am eindeutigsten unter der finanziellen Ägide eines oder mehrerer Unternehmen. Das mobile Betriebssystem *Android* beispielsweise wurde 2007 durch *Google* und die *Open Handset Alliance* lanciert, die Mitte 2015 aus 87 Firmen bestand, darunter vorrangig Mobilfunkbetreiber und Hardwarehersteller. Gängige *Android*-Distributionen funktionieren nur im Verbund mit proprietären Gerätetreibern und sind mit *Google*-eigenen Diensten verschränkt, was die *Europäische Kommission* (2015) zu einer kartellrechtlichen Prüfung anregte, für die kooperierenden Firmen angesichts ihrer divergierenden Interessen aber unproblematisch sein dürfte. Insoweit lässt sich die *Open Handset Alliance* in gewisser Hinsicht als ein klassisches ‚industry consortium' beschreiben (vgl. dazu: Werle 2000), das auf die Schaffung wechselseitig profitabler Standards ausgerichtet ist. Auch das durch *Apple* angestoßene *WebKit*-Projekt

dient primär diesem Zweck. *MySQL* hingegen ist ein Beispiel für ein Produkt mit dualem Lizenzsystem und direkt an *Oracle* angebunden.

Infrastrukturprojekte, deren Produkte übergreifenden Einsatz erfahren, werden ebenfalls vorwiegend durch Unternehmen getragen (vgl. Websites der Stiftungen): Neben der Allokation von Entwicklerressourcen werden die Projekte der *Linux Foundation* über die Jahresbeiträge korporativer Mitglieder finanziert, darunter 200 ‚silver members' (20.000+ US-Dollar), 14 ‚gold members' (100.000+ US-Dollar) und acht ‚platinum members' (500.000+ US-Dollar), denen jeweils einer der 16 Sitze im Verwaltungsrat zugesprochen wird. Ein ähnliches Modell verfolgt die *OpenStack Foundation*, deren Hauptsponsoren (2015: *HP*, *Canonical*, *AT&T*, *IBM*, *Intel*, *Rackspace*, *Red Hat*, *SUSE*) gleichermaßen einen Sitz im Aufsichtsrat erhalten. Im Falle der *Apache Software Foundation*, die für 2013/2014 Zuwendungen von rund 1,1 Mio. US-Dollar ausweist, wird das ‚board of directors' hingegen komplett durch die Mitgliederbasis gewählt. Allerdings zeigt sich, dass dort mit Sam Ruby (*IBM*), David Nalley (*Citrix*) und Rich Bowen (*Red Hat*) ebenfalls Mitarbeiter größerer Softwarehersteller sitzen (Stand: 7/2015). Zu den führenden Geldgebern gehörten Mitte 2015 u. a. *Google*, *Facebook*, *Yahoo!*, *Citrix*, *Cloudera* und *Microsoft*. Die Programmiersprachen *PHP* und *Python* sowie der Anzeigeserver *X.Org* werden gleichermaßen primär durch korporative Spenden finanziert.

Ein Sonderfall im Bereich der quelloffenen Infrastrukturstandards, der auf potenzielle Risiken des Open-Source-Modells hinweist, besteht in der Verschlüsselungssoftware *OpenSSL*, die in vielen gängigen Betriebssystemen (z. B. *Android*, *Debian*, *Amazon Linux*), Softwarearchitekturen (z. B. Banking- bzw. Kreditkartensysteme) und Web-Plattformen (z. B. *Wikipedia*, *Amazon Web Services*, *GitHub*, *Tumblr*) Einsatz findet. Trotz dieser weitläufigen Verbreitung wurde *OpenSSL* bis zur Formierung der *Core Infrastructure Initiative* (vgl. Kap. 3.2) im Wesentlichen durch einen Vollzeitprogrammierer sowie ein kleines, freiwilliges Kernteam entwickelt und erhielt kaum finanzielle Unterstützung durch die Softwareindustrie: “OpenSSL is completely unfunded [...]. It's used by companies who make a lot of money, but almost none of the companies who use it contribute anything at all” (Ben Laurie in Perlroth 2014). Dieses Ungleichgewicht führte dazu, dass anfragenbasiert zwar kontinuierlich neue Features in *OpenSSL* integriert wurden, aber kaum mehr Wartungsarbeiten stattfanden, was 2012 in einem Flüchtigkeitsfehler in der Programmierung mündete, aus dem wiederum eine gravierende Sicherheitslücke (‚Heartbleed') resultierte, die erst 2014 entdeckt wur-

de und das Auslesen geschützter Daten auf nahezu allen informationstechnischen Geräten weltweit ermöglichte (vgl. Stokel-Walker 2014): "'Catastrophic' is the right word. On the scale of 1 to 10, this is an 11" (Schneier 2014).

Aber nicht nur zentrale Infrastrukturvorhaben und explizit von Unternehmen initiierte Projekte, auch gesellschaftsethisch fundierte Entwicklergemeinschaften wie *KDE* und *GNOME* (Desktop-Umgebungen) oder *Debian* (*GNU*/*Linux*-Distribution) rekurrieren in ihrer Grundfinanzierung auf Unternehmungszuwendungen: Die *GNOME Foundation* erhielt 2013 bei Gesamteinnahmen von 272.000 US-Dollar (ohne ‚Outreachy Program') 140.000 US-Dollar von Mitgliedern ihres Advisory Boards (u. a. *Google*, *IBM*, *Intel*, *Red Hat*); der hinter *KDE* stehende Verein nahm 2013 bei Gesamteinkünften von 190.292 Euro rund 77.000 Euro an korporativen Beiträgen und 79.000 Euro durch gesponserte Veranstaltungen ein; *Debian* unterhält Partnerschaften mit einer Vielzahl kleinerer und mittelgroßer Unternehmen und lässt sich Hardware sowie Hosting-Services finanzieren. Und auch die *Free Software Foundation* generierte 2013 über 80 Prozent ihres Umsatzes (1,2 Mio. US-Dollar) durch unternehmerische Zuwendungen.

Aufgrund ihrer Geschichte und Größe nicht in diese taxonomische Trias einordnen lässt sich die *Mozilla Foundation*, die 2003 aus der mit *Netscape*/*AOL* assoziierten *Mozilla Organization* hervorgegangen ist (vgl. Kap. 2.4) und inzwischen über 1000 festangestellte bezahlte Mitarbeiter beschäftigt (vgl. Hardy/Bilton 2014). Für 2013 wies die *Mozilla Foundation* (2014) Einnahmen von 314 Mio. US-Dollar aus (2012: 311 Mio. US-Dollar), davon 97 Prozent aus Lizenzeinnahmen für die Integration spezifischer Suchanbieter in die *Firefox*-Browserleiste, die bis 2013 hauptsächlich aus Vereinbarungen mit *Google* resultierten. Seit 2014 heißt der voreingestellte Suchdienst in den USA hingegen *Yahoo!*. Ab 2014 schaltete *Mozilla* in *Firefox* überdies sogenannte ‚sponsored tiles' (Werbung), annoncierte aber Ende 2015, angesichts kritischen Nutzer-Feedbacks künftig auf diese Einnahmequelle verzichten zu wollen. Mit Blick auf die Zahl fester Mitarbeiter, die finanziellen Kennzahlen und Umsatzstrategien hat *Mozilla* insgesamt nur noch wenig mit einem Open-Source-Vorhaben im eigentlichen Sinne gemein, sondern lässt sich eher mit einem klassischen Anbieter vergleichen.

Die meisten großen Open-Source-Projekte stehen heute insofern in einem finanziellen Wechselverhältnis mit führenden IT-Konzernen, die als Teil ihrer übergreifenden Marktstrategien gezielt in spezifische Entwicklungsvorhaben investieren. Im Falle korporativ initiierter Projekte liegt diese Verschränkung auf der Hand; aber auch stiftungsgetragene Communitys sprechen ihren

Geldgebern Sitze in den ‚boards' ihrer Dachorganisationen zu, welche die Entwicklungsaktivitäten zwar zumeist nicht direkt steuern, aber die technischen Infrastrukturen dafür bereitstellen und finanzielle Ressourcen verteilen. Kombiniert mit ihrem Involvement in die konkrete Code-Entwicklung, sichern sich die jeweiligen Unternehmen so einen nicht zu unterschätzenden Einfluss auf relevante Entwicklungsvorhaben und tragen zugleich zu einer Erhöhung der Planungssicherheit in diesen Projekten bei.

4 Spielarten quelloffener Softwareprojekte

Insgesamt lassen sich vor dem Hintergrund der steigenden sozioökonomischen Relevanz des IT-Sektors im Allgemeinen drei wesentliche Trends in der Entwicklung von Open-Source-Projekten und quelloffener Software herausstellen:

- Zum ersten haben sich ab Ende der 1980er-Jahre sowohl *rechtlich belastbare Lizenzdefinitionen* für quelloffene Softwareprojekte etabliert, welche deren Arbeitsergebnisse wirksam vor Einzelaneignung schützen, als auch übergreifend akzeptierte *Regeln, Werte und Arbeitskonventionen* herauskristallisiert, deren Ausdifferenzierung bzw. Verbreitung durch die kommunikationserleichternden Eigenschaften der Online-Technologien erheblich befördert wurde.
- Zum zweiten sind Open-Source-Komponenten in den letzten 20 Jahren zu einem *integralen Bestandteil des Softwaremarktes* geworden und spielen heute sowohl für die basalen Infrastrukturen des Internets als auch in der Unternehmensinformatik eine zentrale Rolle. Darüber hinaus wirken restriktiv und quelloffen lizenzierte Komponenten nicht nur im Zuge ihrer konkreten Indienstnahme ineinander, sondern werden in vielen Softwarepaketen bewusst miteinander kombiniert.
- Zum dritten lässt sich damit einhergehend eine zunehmende *Korporatisierung und Kommerzialisierung* quelloffener Softwareprojekte beobachten: Etablierte Unternehmen haben sich auf deren institutionelle Rahmenbedingungen eingestellt, nutzen entsprechende Kontexte zur Ergänzung ihrer proprietären Entwicklungsarbeit und zur Kooperation mit Mitbewerbern, protegieren für ihr Geschäft förderliche Projekte und haben neue Dienstleistungsfelder rund um Open Source erschlossen.

Im Horizont dieser Dynamiken hat sich eine große Bandbreite an sehr unterschiedlich ausgerichteten Vorhaben um Open-Source-Software herausgebildet: Am einen Ende des Spektrums finden sich Gemeinschaften, die nach wie vor Richard Stallmans gesellschaftsethischen Maximen verpflichtet sind, unabhängig von korporativen Interessen operieren und sich weitgehend an egalitären Organisationsprinzipien ausrichten; am anderen Ende lässt sich eine Vielzahl an Projekten identifizieren, die hierarchischen Koordinations-

bzw. Entwicklungsmodellen folgen und unter der unmittelbaren Kontrolle großer Technologieunternehmen stehen.

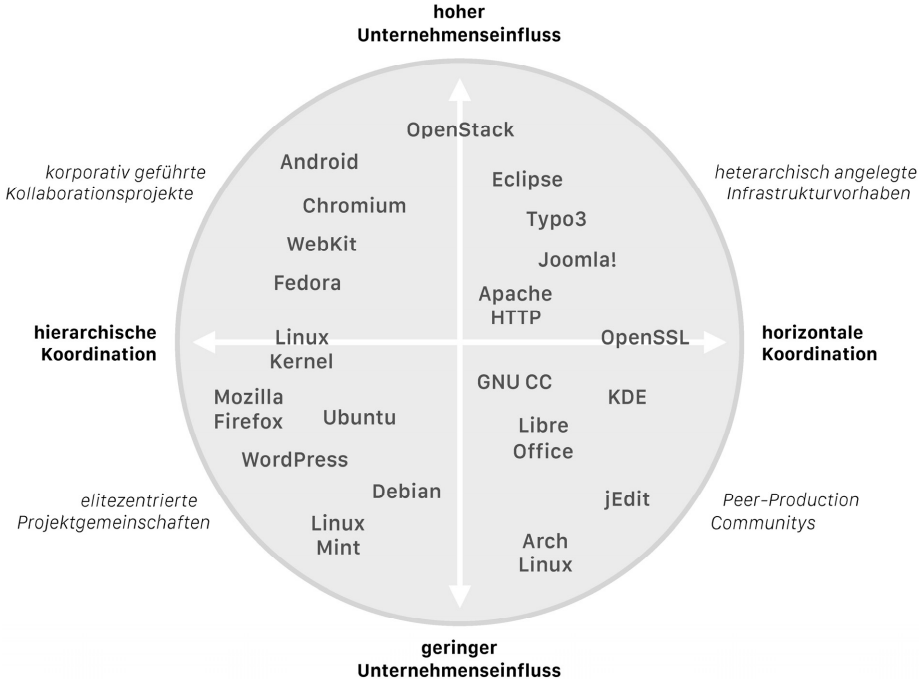

Abb. 9 Idealtypische Varianten quelloffener Softwareprojekte

Auf der Grundlage verfügbaren empirischen Materials (z. B. lizenzrechtliche Dokumente, technische Spezifikationen, Mitgliederlisten, Eigenbeschreibungen, Mailinglisten, Wikis) sowie vorliegender Fallstudien und Übersichtsdarstellungen lassen sich entlang ihrer vorherrschenden Koordinationsweisen und dem Grad ihrer Unternehmensnähe vier typische Spielarten derzeitiger quelloffener Entwicklungsvorhaben voneinander abgrenzen (s. Abb. 9):

(1) *korporativ geführte Kollaborationsprojekte* zur Erarbeitung gemeinsamer Produkte und Plattformen, die durch ein anleitendes Unternehmen gesteuert werden;
(2) *heterarchisch angelegte Infrastrukturvorhaben*, die durch dezentralere Entscheidungsmuster geprägt sind, meist nicht durch kommerzielle Anbieter initiiert wurden, heute allerdings primär auf deren Leistungen basieren;

(3) *elitezentrierte Projektgemeinschaften*, welche sich durch einen moderateren Einfluss spezifischer Unternehmen auszeichnen, aber von eng definierten Kerngruppen geführt werden; und
(4) eine vergleichsweise kleine Zahl an egalitär ausgerichteten *Peer Production Communities* in Yochai Benklers (2002) Sinne, die sich durch eine marktunabhängige Kollaboration unter Gleichberechtigten auszeichnen.

4.1 Korporativ geführte Kollaborationsprojekte

Ein Gutteil der marktrelevanten Open-Source-Projekte steht in einem direkten Verweisungszusammenhang mit marktführenden Technologiekonzernen, die in den letzten 15 Jahren zunehmend eigene Entwicklungsvorhaben angestoßen haben, statt ihre Aktivitäten auf bereits gegebene Projektkontexte zu gründen (vgl. West/Bogers 2014; Riehle 2012). Angesichts der Dominanz von in Unternehmen angestellten Entwicklern unter den Beiträgern steht dabei inzwischen allerdings weniger das Abschöpfen von "voluntary external work" (Schaarschmidt et al. 2015: 100) im Vordergrund, sondern vielmehr die projektbezogene Kollaboration mit anderen industriellen Stakeholdern, um gemeinsam Produkte zu erarbeiten und entsprechende Synergieeffekte ohne die Ausgestaltung formaler Kooperationsbeziehungen zu nutzen.

Im Folgenden werden mit *Android* (Betriebssystem), *WebKit* (Browser) und *Fedora* (*Linux*-Distribution) drei Beispiele für unternehmensgeführte Open-Source-Projekte vorgestellt, bevor *OpenStack* (Cloud-Computing-Architektur) als Mischform aus Infrastrukturvorhaben und korporativ geleiteter Kollaborationsplattform beleuchtet wird. Diese Projekte gehören gemessen an ihrer Aktivität und dem qua Basic Constructive Cost Model geschätzten Arbeitsaufwand (vgl. Boehm 1981; kritisch: Trendowicz/Jeffery 2014) zu den bis dato bedeutsamsten Open-Source-Vorhaben (s. Tab. 8).

Tabelle 8:
Kennzahlen ausgewählter korporativ geführter Kollaborationsprojekte

	geschätzter Aufwand Stand 9/2015, Basic COCOMO*	Commits 5/2014 bis 5/2015	Lizenz-modell	strategische Kontrolle
Android	~ 4.000 Personenjahre	81.119	permissive	*Google / Open Handset Alliance*
Chromium (Browser)	~ 3.000 Personenjahre	61.454	permissive	*Google*
WebKit	~ 1.500 Personenjahre	14.867	permissive	*Apple*
Fedora (Packages & Documentation)	~ 200 Personenjahre	45.232	gemischt	*Red Hat/Council*
OpenStack	~ 700 Personenjahre	62.370	permissive	Sponsoren / Komitee

Quelle: Open Hub. * Basic Constructive Cost Model (organisch [a=2.4, b=1.05])

Android

Android wurde ab 2003 als Betriebsumgebung für Mobile Devices durch ein gleichnamiges Start-up-Unternehmen entwickelt, das 2005 für ca. 50 Mio. US-Dollar von *Google* aufgekauft wurde. Bis 2007 akquirierte *Google* zwei Duzend korporative Partner aus dem Bereich der Gerätehersteller und Netzwerkbetreiber, baute ein "secret patent portfolio" (Claburn 2007) auf und stellte Ende 2007 das Projekt sowie die *Open Handset Alliance* öffentlich vor. Die Entscheidung, *Android* als Open-Source-Vorhaben anzulegen, lässt sich zum einen darauf zurückführen, dass das System auf dem *Linux Kernel* und weiteren quelloffenen Komponenten fußt, die sich ohnehin nicht proprietarisieren lassen. Zum anderen flexibilisiert der Rückgriff auf freie Lizenzen die Zusammenarbeit zwischen den (inzwischen knapp 90) beteiligten Unternehmen, die nach der Präsentation von *Apples iPhone* Anfang 2007 erheblich an Beschleunigung erfahren musste. Dabei ging es *Google* von vornherein weniger darum, mit *Android* selbst Umsatz zu generieren, als den Zugriff auf seine Webdienste und Handelsplattformen zu erweitern, was sich auch in der soziotechnischen Verfasstheit des Projekts zeigt. *Android*-eigener Code steht unter permissiven Lizenzen, welche die Einbindung in kommerzielle Produkte erleichtern und *Google* gleichzeitig weitreichende Steuerungsmöglichkei-

ten einräumen, die sich beispielsweise im ‚Contributor Agreement' wie folgt widerspiegeln:

> "You hereby grant to the Project Leads and to recipients of software distributed by the Project Leads a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare derivative works of, publicly display, publicly perform, sublicense, and distribute your contributions and such derivative works." (Google 2015b)

Google ist als Projektleiter hingegen lediglich dazu verpflichtet, den Source-Code finalisierter Versionen zu publizieren, und kann so im Verbund mit weiteren Rahmensetzungen wie der ‚Compatibility Definition' (Google 2015c) die technischen Spezifikationen abstecken, denen die weitere Entwicklung und Produktumsetzung folgt. So kontrollieren die integrierten Programmierschnittstellen nicht nur den Zugriff auf spezifische Funktionen, sondern definieren darüber hinaus die technischen Eigenschaften von *Android*-Geräten mit und legen die Einbindung *Google*-eigener Services nahe. Dieses Top-down-Management wirkt einerseits einer Fragmentierung der Architekturen entgegen und erhöht die Erwartungssicherheit, was zu der raschen Verbreitung des mobilen Betriebssystems beigetragen hat. Andererseits wird *Android* durch diese zentralisierten Entscheidungsmuster aber zugleich zu dem bislang "most closed open source project" (Vision Mobile 2011). Auch *Chromium* als das zweite große von *Google* initiierte Open-Source-Projekt wird unmittelbar durch das Unternehmen kontrolliert und verfügt über keine vollständig einsehbaren Koordinationsstrukturen.

WebKit

WebKit ist eine 2001 durch *Apple* angestoßene Abspaltung der von der *KDE*-Community entwickelten *KHTML Engine* zur Darstellung von Web-Inhalten und die Basis für *Apples* Browser *Safari* in *OS X* und *iOS*, zahlreiche gerätespezifische Browser (z. B. *Sony PlayStation*) sowie Browser in *Linux*-Umgebungen. Ähnlich wie *Android* steht *WebKit* gemessen an regelmäßig aktiven Entwicklern und Reviewern eindeutig unter der Ägide seines Gründungsunternehmens, das auch die regelmäßigen ‚contributors meetings' ausrichtet, verfügt im Unterschied dazu aber nur über wenige prädefinierte Richtlinien und dokumentiert öffentlich, welche Entwickler aus welchen Firmen (u. a. *Samsung*, *Google*, *Intel*) an welchen Programmierarbeiten beteiligt sind (vgl. Teixeira/Lin 2014; Bitergia 2013). Hinter *WebKit* steht zudem kein festgeschriebenes Firmenkonsortium; die beteiligten Entwickler arbeiten vielmehr sachbezogen auf individueller Ebene zusammen an einer Softwarearchitek-

tur, die in zahlreiche herstellerspezifische Produkte eingeht. “Maybe this is the reason why it can work, with such as deep involvement of fiercely competing companies: letting developers discuss without representing companies help to keep decisions technical and neutral” (Gonzalez-Barahona/ Robles 2013: 178). Allerdings lassen sich auch im Falle von *WebKit* keine voll ausdefinierten Entscheidungswege oder formal bestimmten Steuerungsgruppen identifizieren; das Projekt richtet sich vielmehr an seinen erfahrensten Entwicklern aus, die in aller Regel bei *Apple* angestellt sind.

Fedora

Von solchen konzernseitig initiierten Vorhaben hebt sich auf den ersten Blick die verbreitete *Linux*-Distribution *Fedora* ab, die 2002 im Rahmen eines studentischen Informatikprojektes entstanden ist und im Unterschied zu *Android* oder *WebKit* über ein gewähltes technisches Komitee verfügt, welches Richtungsentscheidungen trifft. Auch *Fedora* ist jedoch eng mit einem auf seinem Feld marktführenden Unternehmen verknüpft, obgleich sich *Red Hat* gemessen an seinem Jahresumsatz (2014: 1,5 Mrd. US-Dollar) nicht mit *Apple* oder *Google* vergleichen lässt (vgl. Kap. 3.3): 2003 beschloss *Red Hat*, seine *Linux*-Distribution in eine kommerzielle Version (*Red Hat Enterprise Linux*) und eine freie Variante aufzuspalten, die in das *Fedora*-Projekt überführt wurde. 2005 war *Fedora* kurzzeitig ein stiftungsgetragenes Vorhaben; da allerdings das US-amerikanische Steuerrecht der Unterstützung der Stiftung durch *Red Hat* enge Grenzen setzte, entschied sich das Unternehmen bereits 2006 dazu, die *Fedora Foundation* wieder aufzulösen. Seitdem gehört *Fedora* juristisch zu *Red Hat* und wird durch ein ‚board‘ bzw. seit 2014 durch ein konsensorientiertes ‚council‘ geführt, das sich zu zwei Dritteln aus gewählten Mitgliedern und zu einem Drittel aus *Red Hat*-Mitarbeitern zusammensetzt, darunter auch der vetoberechtigte Projektleiter. Richtlinienentscheidungen werden öffentlich einsehbar durch ein gewähltes technisches Steuerungskomitee getroffen. Die Teilnehmerbasis des Projektes speist sich neben Studierenden vorrangig aus bei *Red Hat* angestellten Entwicklern und korporativen Nutzern von *Red Hat Enterprise Linux*. Finanziert wird *Fedora* beinahe vollständig durch das Unternehmen *Red Hat*, das regelmäßig *Fedora*-Events ausrichtet.

OpenStack

Die 2010 u. a. durch die *NASA* und das Webhosting-Unternehmen *Rackspace* angestoßene Entwicklung der Cloud-Computing-Architektur *OpenStack* hin-

gegen wird durch kein Einzelunternehmen gesteuert, sondern durch die Beiträge von mittlerweile über 130 involvierten Firmen getragen, die (im Gegensatz zu *WebKit*) auf ausdrücklicher Unternehmensebene miteinander kooperieren. Formal werden die Rahmenbedingungen des Projekts durch die 2012 gegründete *OpenStack Foundation* definiert; auf Arbeitsebene allerdings verteilt sich die Kontrolle der Entwicklungsarbeit neben einem gewählten Komitee auf die langfristig aktivsten industriellen Stakeholder, die als Platin-Sponsoren auch das Steuerungsboard der Stiftung dominieren, darunter *HP*, *IBM*, *Intel*, *AT&T* und *Rackspace* sowie die *Linux*- Distributoren *Red Hat*, *Canonical* und *SUSE*. Insofern lässt sich *OpenStack* als eine im Vergleich zu *Android* deutlich horizontaler organisierte ‚community of companies' charakterisieren, in der regelmäßig aktive Unternehmen indes eine privilegierte Rolle einnehmen. "This new kind of community [...] is clearly driven by corporate interests. Participating companies, which may be commercial competitors, have clear strategies towards the project [...]" (Gonzalez-Barahona et al. 2013: 39). Einige Beobachter diagnostizieren insofern einen „harten Konkurrenzkampf" (Grüner 2015) unter den beteiligten Unternehmen, die zuweilen für sie wesentliche personelle und finanzielle Ressourcen in das Projekt investieren. Zudem ist *OpenStack* in der Praxis einzig mit kommerziellen Zusatzprodukten einsatzfähig, was sich auch auf die bislang eher zurückhaltenden Vereinheitlichungsbemühungen der *OpenStack Foundation* zurückführen lässt.

In allen betrachteten Fällen setzen sich die Projektgemeinschaften vorrangig aus angestellten Entwicklern zusammen, die problemzentriert auf individueller Ebene oder als explizite Vertreter ihrer Unternehmen zusammenarbeiten. Eine solche korporative Kollaboration unter Open-Source-Lizenzmodellen trägt zur Überwindung zweier *knowledge sharing dilemmas* (vgl. Larsson et al. 1998; Cabrera/Cabrera 2002) im unternehmerischen Bereich bei: Zum einen verhindern quelloffene Lizenzen die Einzelaneignung des erarbeiteten Codes; zum anderen stellen sich diese Lizenzen Trittbrettfahrern entgegen, welche von den Resultaten kollektiven Engagements profitieren wollen, ohne selbst Ressourcen einzubringen, denn in Open-Source-Projekten bleibt es stets nachvollziehbar, welche Firmen sich auf welche Elemente stützen und inwieweit sie an deren Entwicklung partizipieren (vgl. Henkel et al. 2014; Bogers 2012). Zudem liegt es in der Schöpfung von Softwareprodukten heute ohnehin oft näher, anstatt einer kompletten Neuentwicklung auf quelloffenen Architekturen aufzubauen. Eine Finanzierung von Open-Source-

Projekten kann im Zweifel erheblich günstiger sein als die Lizenzierung proprietärer Lösungen oder die Verletzung von Schutzrechten.

Die konkrete Entwicklungsarbeit erfolgt in den diskutierten Entwicklungsvorhaben auf eigenen technischen Plattformen in Kombination mit Standardtools wie der Versionsverwaltung *Git* oder dem Fehlermeldungssystem *Bugzilla*, welche die ortsungebundene Zusammenarbeit vereinfachen. Korporativ geführte Kollaborationsprojekte haben gleichwohl nicht mehr viel mit dezentral strukturierten Communitys gemein, sondern zeichnen sich durch prägnante Hierarchisierungen aus. Zwar finden modulare und iterative Methoden in allen hier diskutierten Vorhaben Anwendung; welche Änderungen jeweils in die Finalversionen eingehen und welchen übergreifenden Schwerpunkten das Projekt folgt, definieren aber meist nur wenige Entscheidungsträger (vgl. Fjeldstad et al. 2012; Capiluppi et al. 2013). Im Falle von *Android*, *WebKit* und *Fedora* liegt die strategische Kontrolle eindeutig bei den Einzelunternehmen *Google*, *Apple* und *Red Hat*; im Falle von *OpenStack* haben große IT-Unternehmen wie *Hewlett-Packard* oder *IBM*, die mit sehr vielen Programmierern in die Entwicklung involviert sind und über die finanziellen Mittel verfügen, um als zentrale Sponsoren aufzutreten, ebenfalls einen steuernden Einfluss auf das Projekt.

4.2 Heterarchisch angelegte Infrastrukturvorhaben

Von korporativ geführten Projekten heben sich graduell gewachsene Vorhaben ab, die auf grundsätzliche softwaretechnische Standards und Infrastrukturen fokussieren, eher horizontalen Organisationsmustern folgen und sich heute zwar ebenfalls auf die Leistungen etablierter IT-Unternehmen stützen, aber nicht deren direkter Kontrolle unterliegen. Sie werden in der Regel durch Stiftungen getragen und weisen sich mit der Zeit oft deutlich wandelnde Beteiligungsmuster auf (vgl. Wasserman 2013). In solchen heterarchisch angelegten Infrastrukturvorhaben geht es weniger um ‚sichtbare' Produkte als um die Entwicklung übergreifend eingesetzter Werkzeuge, Frameworks und Bibliotheken – von integrierten Entwicklungsumgebungen wie *Eclipse* über Content-Management-Systeme wie *Typo3* oder *Joomla* bis hin zu Webserver-Software wie etwa dem *Apache HTTP Server* und Verschlüsselungsprotokollen wie *OpenSSL*. Die meisten dieser Projekte existie-

ren bereits seit weit über einem Jahrzehnt, zeichnen sich durch eine stabilisierte Code-Basis aus und weisen daher inzwischen mitunter ein geringeres Aktivitätsniveau als jüngere Vorhaben auf (s. Tab. 9).

Tabelle 9: Kennzahlen ausgewählter Infrastrukturvorhaben

	geschätzter Aufwand 9/2015, Basic COCOMO*	Commits (5/2014 bis 5/2015)	Lizenz-modell	strategische Kontrolle
Eclipse**	~ 1.800 Personenjahre	3.515	weakly protective	Foundation Board
TYPO3	~ 1.100 Personenjahre	3.381	strongly protective	*TYPO3 Association*
Joomla	~ 120 Personenjahre	3.266	strongly protective	*Open Source Matters Inc.*
Apache HTTP Server	~ 500 Personenjahre	2.356	permissive	Foundation Board
OpenSSL	~ 120 Personenjahre	1.467	permissive	Core Team

Quelle: Open Hub, eigene Recherchen. * Basic Constructive Cost Model (organisch [a=2.4, b=1.05]); ** Teilprojekte Platform / Web Tools Platform

Eclipse

Eclipse ist eine integrierte Entwicklungsumgebung für verbreitete Programmiersprachen (u. a. *Java*, *C*, *C++*, *PHP*), die ursprünglich auf der ab 1984 von *IBM* vertriebenen Plattform *VisualAge* fußt. Das 2001 unter freie Lizenz gestellte Projekt wurde zunächst durch ein von *IBM* angeleitetes Konsortium geführt, bis die Kontrolle 2004 an die *Eclipse Foundation* übergegeben wurde, die für 2014 rund 220 korporative Mitglieder und Einnahmen von 4,3 Mio. US-Dollar ausgewiesen hat, die sich vor allem aus Mitgliedsbeiträgen speisen und der Bezahlung der festen Mitarbeiter der Stiftung dienen (vgl. Eclipse 2015a). In ihrem achtzehnköpfigen Steuerungsboard sitzen neben vier gewählten Mitgliedern wechselnde Vertreter größerer Unternehmen (derzeit u. a. *IBM*, *SAP*, *Oracle*, *Google*, *Red Hat*), die Einfluss auf die Gesamtausrichtung des Projekts nehmen und seit 2015 über "targeted corporate contributions" (Eclipse 2015b) auch spezifische Teilvorhaben fördern können. Trotz dieser Unternehmensnähe versteht sich *Eclipse* als offene meritokratische Gemeinschaft ("the more you contribute the more responsibility

you will earn"), die Wert auf transparente Entscheidungsmuster legt. Neben einsehbaren Sitzungsprotokollen und Mailinglisten lassen sich über das Dashboard (dashboard.eclipse.org) sämtliche Daten zu Aktualisierungen und Mitgliederaktivitäten abrufen. Die technische Entwicklung erfolgt in Teams, die sich eigenständig entlang gewählter Entscheidungsträger strukturieren. In Konfliktfällen schaltet sich die *Eclipse Management Organization* ein, die aus Stiftungsmitarbeitern besteht.

Typo3 und Joomla

Auch die Content-Management-Systeme (CMS) *Typo3* und *Joomla* können auf (teil)proprietäre Vorläufer zurückblicken und werden durch die Beiträge unternehmensaffiliierter Entwickler vorangetrieben: *Typo3* wurde ab 1998 durch Kasper Skårhøj zunächst im Alleingang entwickelt und unter anderem aus religiöser Überzeugung sowie pragmatischen Gründen – "[...] I couldn't create the quality I wanted" (Skårhøj in Phlow 2003) – nach einer kurzen kommerziellen Phase unter freie Lizenz gestellt. *Joomla* entstand 2005 als Abspaltung des ab 2000 durch die Firma *Miro* unter dualer Lizenz vertriebenen *Mambo CMS*, dessen Kernentwickler sich nach einem Konflikt mit *Miro* dafür entschieden, das Projekt eigenständig unter neuem Namen weiterzuführen. Zu diesem Zweck wurde das Non-Profit-Unternehmen *Open Source Matters* (Einnahmen 2014: 0,45 Mio. US-Dollar; primär aus Sponsorships) gegründet, das wie die 2004 als Verein konstituierte *Typo3 Association* (Einnahmen 2014: 0,85 Mio. Euro; primär aus Mitgliederbeiträgen) für die Sicherung der Rahmenbedingungen zuständig ist, aber nicht in die konkrete technische Entwicklung eingreift (vgl. OSM 2015; T3A 2015). Während sich *Typo3* als formal auch so gefasste "community of commercial actors" (Westenholz 2007: 11; vgl. Westenholz 2012) charakterisieren lässt, arbeiten die zumeist ebenfalls unternehmensaffiliierten Entwickler in *Joomla* auf individueller Ebene zusammen. Sowohl *Joomla* als auch *Typo3* erfahren vor allen Dingen durch kleinere und mittelgroße Unternehmen Unterstützung, die im Bereich Webdesign tätig sind und das CMS an ihre geschäftlichen Erfordernisse anpassen wollen. Beide Vorhaben sind in ihrer Grundanlage heterarchisch entlang von weitgehend eigenständig agierenden Arbeitsgruppen organisiert, verfügen aber inzwischen über gewählte herausgehobene Führungsebenen für die übergreifende Projektkoordination.

Apache HTTP Server

Der *Apache HTTP Server* wurde 1994 von einer zunächst kleinen Gruppe an Programmierern als Erweiterung der in universitären Kontexten entstandenen *NCSA HTTPd*-Architektur entworfen, die mit den gegebenen Lösungen unzufrieden waren: "[...] in fact, the absence of a good commercial alternative was a powerful motivation for the launching of the project" (Lerner/ Tirole 2002: 208). Aufgrund seiner permissiven Lizenz erfuhr der *Apache HTTP Server* schnelle Verbreitung und ist früh zu einem zentralen Bestandteil des Softwaremarktes geworden (vgl. Greenstein/Nagle 2014). Zudem galt das Projekt rasch als Paradebeispiel für eine onlinebasierte selbstgesteuerte Kollaboration unter Freiwilligen (vgl. z. B. auch in Benkler 2002). Bereits 1998 ging *Apache* freilich eine erste Partnerschaft mit IBM ein, in der es vorrangig um den Austausch von Wissen ging (vgl. McHugh 1998). Mittlerweile gehören zu der rund 130 Personen starken Kerngruppe des *Apache HTTP Server*-Projekts primär Entwickler aus mittleren und großen IT-Firmen (z. B. *IBM*, *HP*, *Fujitsu*). Daneben genießen der *HTTP Server* und weitere Projekte über die *Apache Software Foundation* finanzielle Unterstützung durch marktführende Unternehmen wie *Google*, *Facebook* und *Microsoft*. Trotz dieses intensiven korporativen Engagements haben sich die Stiftung und ihre Projekte operative Unabhängigkeit bewahrt, was sich auch auf ihre frühzeitig ausgebildeten eigenständigen Koordinationsstrukturen zurückführen lässt (vgl. Di Tullio/Staples 2013): Während das Steuerungsboard der Stiftung durch ihre Mitglieder gewählt wird, die diesen Status einzig durch Qualifikation erlangen können, werden Positionen in der Entscheidungshierarchie des *Apache HTTP*-Projekts leistungsbezogen vergeben – von einfachen Beiträgern über ‚committers', die über Aktualisierungen des Codes abstimmen dürfen, bis hin zum ‚project management committee', das die Endredaktion übernimmt und strategische Entscheidungen trifft (vgl. Apache 2015a).

OpenSSL

Über solche ausdefinierten Koordinationsstrukturen verfügt die hinter der verbreitet eingesetzten Verschlüsselungssoftware *OpenSSL* stehende Projektgemeinschaft nicht. *OpenSSL* galt lange trotz seiner ubiquitären Nutzung als Beispiel für ein basarartig strukturiertes Vorhaben, das weitgehend ohne korporative Förderung operiert. Die 2014 entdeckte Sicherheitslücke ‚Heartbleed' führte allerdings vor Augen, dass das Projekt seiner Marktrelevanz bereits seit geraumer Zeit nicht mehr gewachsen war (vgl. Perlroth 2014):

Seit 1999 wurde *OpenSSL* fast ausschließlich durch den einzigen angestellten Vollzeitmitarbeiter Stephen Henson sowie drei freiwillige Helfer aktualisiert, die nach wie vor ohne ausformulierte Regeln über Mailinglisten zusammenarbeiten. Erst seit 2014 wird *OpenSSL* über die *Core Infrastructure Initiative* in einem signifikanten Umfang durch die Softwareindustrie unterstützt (vgl. Kap. 3.4) und befindet sich insoweit auf dem Weg zu einem korporativ geförderten Infrastrukturprojekt. Gemessen an den auf der Plattform *Open Hub* aggregierten Aktivitätsdaten fußt das Vorhaben allerdings nach wie vor auf der Arbeit eines relativ kleinen Kernteams: Zwischen 9/2014 und 9/2015 finden sich 15 bis 20 kontinuierlich involvierte Entwickler in den Statistiken – das ist nicht viel für ein branchenzentrales Projekt.

Die skizzierten Infrastrukturprojekte zeichnen sich in mehrfacher Hinsicht durch ihre Verwobenheit mit korporativen Kontexten aus: Entweder sie basieren in ihrem Ursprung auf freigegebenen herstellerspezifischen Architekturen (z. B. *Eclipse*) oder sie waren aus sich selbst heraus durch ein rasches Wachstum gekennzeichnet (z. B. *Apache*), da sie Lösungen für zuvor nicht probat adressierte Bereiche boten, und aus diesem Grund früh für Unternehmen interessant. Heute werden *Eclipse*, *Joomla*, *Typo3* und der *Apache HTTP Server* primär durch das Engagement mittlerer und großer IT-Firmen getragen; ihre Communitys werden jedoch im Unterschied zu expliziten Kollaborationsprojekten nicht durch einen korporativen Kernzirkel angeleitet, sondern operieren unter dem Dach gemeinnütziger Organisationen und sind horizontaler angelegt. Gleichwohl haben sich auch in diesen Projekten mit wachsender Größe und Marktrelevanz stufenartige Führungsstrukturen herausgebildet, die der Qualitätssicherung dienen und die Gemeinschaft im Falle divergierender Partikularinteressen zusammenhalten. Die übergeordneten Funktionsträger werden zumeist meritokratisch bzw. leistungsbezogen designiert; allerdings können sich von Unternehmen für das Projekt freigestellte Entwickler in der Regel intensiver als Freizeitprogrammierer in die Community einbringen und so eher Entscheidungspositionen erlangen.

4.3 Elitezentrierte Projektgemeinschaften

Elitezentrierte Projektgemeinschaften zeichnen sich ähnlich wie viele Infrastrukturvorhaben durch ein organisches Wachstum aus und stehen nicht unter

der Ägide eines etablierten Unternehmens. Vielmehr werden Projekte wie der *Linux Kernel*, *Debian* und *Ubuntu* (*Linux*-Distributionen), *WordPress* (Content-Management-System) oder *Mozilla Firefox* (Browser) durch einen eng definierten Führungszirkel angeleitet, an dessen Spitze oft ihr Gründer steht. Auch wenn die Zentralstellung von Einzelpersonen der Idee einer selbstgesteuerten Kollaboration unter Gleichberechtigten entgegensteht, könnten – so argumentiert zumindest Raymond (2000) – Projektleiter in Open-Source-Kontexten nicht despotisch agieren, sondern müssten stets das Wohl der Gemeinschaft im Blick haben, da deren Entwickler andernfalls auf der Basis der gegebenen quelloffenen Architekturen jederzeit neue Gemeinschaften gründen könnten. Benkler (2013: 225) vertritt zudem die Meinung, dass technische Plattformen wie die Versionsverwaltung *Git* einer "internal ossification of power" entgegenwirken: "The technical infrastructure as it develops is supporting more distributed models and providing easier pathways to challenge oligarchy [...]." Für die nachfolgend thematisierten Entwicklungsvorhaben lässt sich ein solcher technikinduzierter Einflussverlust ihres Führungspersonals allerdings kaum diagnostizieren (s. Tab. 10).

Tabelle 10: Kennzahlen ausgewählter elitezentrierter Projektgemeinschaften

	geschätzter Aufwand 9/2015, Basic COCOMO*	Commits (5/2014 bis 5/2015)	Lizenz-modell	strategische Kontrolle
Linux Kernel	~ 5.900 Personenjahre	65.406	strongly protective	L. Torvalds
Ubuntu**	~ 760 Personenjahre	32.485	(strongly) protective	M. Shuttle-worth
Linux Mint	~ 140 Personenjahre	1.439	(strongly) protective	C. Lefebvre / Team
Debian	~ 24.800 Personenjahre	21.076	(strongly) protective	jährlich gewählter Projektleiter
WordPress	~ 100 Personenjahre	3.621	strongly protective	M. Mullenweg
Mozilla Firefox	~ 4.200 Personenjahre	60.226	protective / permissive	M. Baker / Board

Quelle: *Open Hub*; eigene Recherchen. * Basic Constructive Cost Model (organisch [a=2.4, b=1.05]); ** inklusive *Ubuntu Touch*

Linux Kernel

Der *Linux Kernel* ist in den letzten zwei Jahrzehnten zu der Grundlage zahlreicher Infrastrukturen und Produktlinien geworden – von Flachbildfernsehern, über Smartphones bis hin zu Bankautomaten. Im Gegensatz zu korporativ geführten Projekten ist das früh zum Aushängeschild für Open-Source-Software avancierte Vorhaben nicht durch ein Unternehmen lanciert worden; es wird heute aber mittlerweile vorrangig von angestellten Entwicklern getragen, darunter zuvorderst Mitarbeiter der IT-Konzerne *IBM*, *Intel* und *Samsung* (vgl. Kap. 2.4). Stabilität erfährt das Projekt – neben wenigen festgeschriebenen Regel und Richtlinien sowie der administrativen Unterstützung der wiederum vorrangig durch unternehmerische Spenden getragenen *Linux Foundation* – durch "a lieutenant system built around a chain of trust" (Kernel.Org 2015a), an dessen Spitze der Projektgründer als "benevolent dictator" (Rivlin 2003) steht: "Linus Torvalds is the final arbiter of all changes accepted into the Linux kernel" (Kernel.Org 2015b). Am Ende des gerade zitierten Dokumentationsartikels wird zum Beispiel schlicht auf einen informellen Mailinglistenbeitrag Torvalds' verwiesen, in dem er einen Vorschlag zur Änderung der Betreffzeile für Korrekturen ablehnt und sein "canonical email format" definiert. Linus Torvalds Managementstil wird als konsensorientiert, aber entscheidungsbereit beschrieben (vgl. Epstein 2015; Li et al. 2012). Der Source-Code ist für alle finalisierten und sich derzeit in Entwicklung befindlichen Versionen frei zugänglich; technische Diskussionen sind über die *Linux Kernel Mailinglist* (ca. 500 Beiträge/Tag) einsehbar.

Ubuntu und Mint

Ähnlich eindeutig sortiert sind die Einflussverhältnisse in den Projekten *Ubuntu* und *Mint*, deren Distributionen 2014 die meistgenutzten *Linux*-Varianten auf dem Desktop waren (vgl. DistroWatch 2015).

Ubuntu wurde 2004 durch Mark Shuttleworth gegründet, der 1999 durch den Verkauf eines Start-up-Unternehmens zum Multimillionär geworden ist, als "self-appointed benevolent dictator for life (SABDFL)" heute "a happily undemocratic role as sponsor of the project" (Ubuntu 2015) spielt und in dieser Funktion die Mitglieder des ‚technical boards' benennen kann, in Gremien eine übergeordnete Stimme hat und Richtungsentscheidungen trifft. Begründet wird dies wie folgt:

> "This is not a democracy, it's a meritocracy. We try to operate more on consensus than on votes [...]. However, it is not uncommon in the open-source world for there to be multiple good arguments, no clear consensus [...]. In many cases, there

> is no one 'right' answer, and what is needed is a decision more than a debate. The SABDFL acts to provide clear leadership on difficult issues, and set the pace for the project." (Ubuntu 2015)

Zudem ist das Shuttleworth direkt unterstehende ca. 700 Mitarbeiter starke Unternehmen *Canonical* (Stand: 11/2015), das seinen Umsatz primär mit Dienstleistungen um *Ubuntu* macht, neben ca. 500 freiwilligen Entwicklern intensiv in das Projekt involviert.

Auch das 2006 gegründete und von rund 20 überwiegend ehrenamtlich tätigen Entwicklern getragene *Linux Mint*-Projekt, hinter dem weder eine Firma noch eine Stiftung steht, richtet sich an den Entscheidungen seines Gründers Clement Lefebvre aus, der zwar rege Diskussionen schätzt, aber einen ebenso großen Wert auf starke Führung legt: "The final decision comes from the top [...]. Strong leadership is important and benefits Linux Mint, [because] the decisions we take remain consistent and are coherent with our overall vision" (Lefebvre in Byfield 2013; vgl. Eitzen 2013).

Debian

Im Unterschied dazu steht im *Debian*-Projekt ein demokratisch bestimmter Projektleiter an der Spitze, der befristet ähnliche Befugnisse wie Shuttleworth oder Torvalds hat. *Debian Linux* wurde 1993 durch den Studenten Ian Murdock initiiert, verfügte 2014 über ca. 1000 freiwillige Entwickler (vgl. Perrier 2014), gehört mit einem Marktanteil von zwölf Prozent zu den verbreiteten *Linux*-Distributionen im Serverbereich und erhält neben Kleinzuwendungen über die NPO *Software in the Public Interest* (2015) primär Ressourcenspenden von mittelgroßen IT-Unternehmen. Seit 1997 verfügt *Debian* über einen Gesellschaftsvertrag, eine Verfassung und einen im Jahresturnus neu gewählten Projektleiter (Wahlbeteiligung 2015: 34 Prozent), der – ähnlich wie ein CEO – das interne Projektmanagement übernimmt, die Gemeinschaft nach außen repräsentiert und deren ‚Vision' definiert sowie Mitglieder technischer Ausschüsse berufen, Ressourcen verteilen und ggf. Ad-hoc-Entscheidungen treffen kann (vgl. Debian 2015; Roeckx 2015). Seine Befugnisse sind das Resultat jahrelanger Aushandlungskontroversen (vgl. Coleman 2013; O'Mahony/Ferraro 2007). Unterhalb dieser Führungsebene lässt sich *Debian* als ‚bazaar of cathedrals' (vgl. Krafft 2010) beschreiben, in dem delegierte Entwickler jeweils eigenständige Module betreuen. Neumitglieder müssen ein Prüfungsverfahren hinsichtlich ihrer Kompetenz und Verbundenheit mit den Projektleitlinien durchlaufen, darunter auch die Maxime, keine unfreie Software in das System einfließen zu lassen. Diese ideologi-

sche Fundierung gilt als ein Grund für die geringe Durchdringung des Projekts mit unternehmensaffiliierten Entwicklern.

WordPress

Ausgeprägte institutionelle Strukturen lassen sich auch in der *WordPress*-Community finden, die jedoch wiederum sehr gründerzentriert ist: *WordPress* wurde 2004 durch den Studenten Matt Mullenweg und den Web-Entwickler Mike Little veröffentlicht, ist heute das meistgenutzte Content-Management-System im Netz und wird durch ein zwanzigköpfiges Kernteam sowie eine meritokratisch strukturierte Projektgemeinschaft unter der alleinigen Leitung von Mullenweg aktualisiert. Anders als der *Linux Kernel* verfügt *WordPress* über mehrere ‚contributor handbooks', die Auskunft über die Richtlinien des Projekts geben. Mullenweg ist darüber hinaus Vorsitzender der *WordPress Foundation*, welche die weltweit stattfindenden *WordCamps* (2014 ca. 75 Veranstaltungen) ausrichtet, und CEO der 2005 durch ihn gegründeten Start-up-Firma *Automattic*, die 2014 ca. 350 ‚remote workers' beschäftigte und ihren Umsatz mit kostenpflichtigen Zusatzprodukten, Freemium-Services sowie Content-Management für Unternehmen macht. Darüber hinaus ist Mullenweg Gründer der Investmentgesellschaft *Audrey Capital*, die seit 2008 in über 50 IT-Firmen investiert hat, darunter z. B. auch *MakerBot Industries* (3D-Drucker). Angesichts ihrer zentralisierten Entscheidungsstrukturen vermutet der Technikjournalist Shane Snow (2014), dass alle genannten Firmen und Projekte im Falle eines Ausstiegs oder Ablebens ihres Gründers zeitnah eingehen würden: "Despite the dictatorial nature of his management, it's Mullenweg's genuineness and vision that keeps WordPress's contributors contributing, Automattic's employees from churning."

Mozilla Firefox

Das höchste Maß an Top-down-Management unter den nicht unternehmensgeführten Projekten weist *Mozilla Firefox* auf. Im Gegensatz zu anderen Open-Source-Stiftungen nimmt die *Mozilla Foundation* über ihre Tochterfirma mit über 1000 Angestellten direkten Einfluss auf die Projektarbeit: "The Mozilla Corporation [...] works with the community to develop software that advances Mozilla's principles" (Mozilla 2015a). Mit der Publikation des *Firefox*-Browsers 2004 stellte *Mozilla* überdies auf ein "rather rigorously controlled model" (Stamelos 2014: 328) mit ausgeprägten Entscheidungshierarchien um – von ‚super-reviewers' über ‚release drivers' und ‚stewards' bis hin zu zwei ‚ultimate decision makers' (vgl. Mozilla 2015b). Eine

dieser ultimativen Positionen nimmt seit 1998 die ehemalige *Netscape*-Managerin Mitchell Baker ein, die zudem auch CEO der *Mozilla Corporation* und Vorsitzende der *Mozilla Foundation* ist; die zweite belegte bis 2014 der davor ebenfalls für *Netscape* tätige Brendan Eich. Führungsrollen in der Community werden leistungsorientiert vergeben; Entscheidungswege und Sitzungsprotokolle sind öffentlich einsehbar. Ein zentraler Orientierungspunkt ist das *Mozilla Wiki*, in dem auch kritische Punkte wie die Koordinationsprobleme zwischen angestellten Entwicklern und den nach eigenen Angaben mehr als 10.000 Freiwilligen thematisiert werden. In den ‚weekly updates' (wiki.mozilla.org/WeeklyUpdates) wurden zwischen 1/2015 und 7/2015 freilich über 95 neue Mitarbeiter, aber keine neuen Freiwilligen vorgestellt. Von den 314 Mio. US-Dollar Einnahmen im Jahr 2013 gab die *Mozilla Foundation* 65 Prozent für Entwicklung und 25 Prozent für Marketing und Verwaltung aus. Nicht nur hinsichtlich dieser Verteilungen, auch angesichts der hierarchischen Strukturen in der Stiftung und ihren Projekten folgt *Mozilla* eher klassischen unternehmerischen Organisationsmustern.

Sowohl der *Linux Kernel*, *WordPress* und *Mozilla* als auch *Linux Mint* und *Ubuntu* sind in ihren Entscheidungsabläufen auf ihre Gründer bzw. langjährigen Manager ausgerichtet, während die *Debian*-Community jährlich einen Projektleiter wählt. Kritische Entschlüsse werden durch diese Führungspersonen im Verbund mit einem kleinen Kernzirkel gefasst, wobei alle genannten Projekte auch unterhalb dieser Steuerungselite über herausgehobene Entscheidungsträger verfügen, die vereinheitlichend tätig werden sowie Maßnahmen zur Qualitätssicherung ergreifen. All dies beschneidet die Spielräume der beteiligten Entwickler, bietet aber auch Orientierung und wirkt einer Fragmentierung der Projekte entgegen. Im Falle von *Debian*, *Mozilla* und *WordPress* sind deren tragende Richtlinien formal fixiert worden; im Falle von *Mint* und dem *Linux Kernel* haben sich entlang des Führungsstils ihrer Gründer hingegen lediglich "opaque governing norms" herausgebildet, die letztlich der proklamierten Offenheit in diesen Projekten entgegenlaufen: "[...] without the law or a clear mechanism of accountability those injured by or excluded from peer production processes have very limited recourse" (Kreiss et al. 2011: 252). Insoweit sind es soziale Grundordnungen und nicht technische Strukturen per se, die bestenfalls einer "internal ossification of power" (Benkler 2013: 225) entgegenwirken können.

4.4 Peer-Production-Communitys

Von den diskutierten Varianten lassen sich schließlich von korporativen Interessen weithin abgekoppelte Entwicklergemeinschaften unterscheiden, die sich durch "voluntaristic, nonstate, nonmarket action" bzw. "nonproprietary, [...] self-organized practices" (Benkler 2013: 213, 230) auszeichnen, und deren Teilnehmer sich primär aus intrinsischen Motiven einbringen: "Basically, people who participate in peer production communities love it. They feel passionate about their particular area of expertise and revel in creating something new or better" (Tapscott/Williams 2006: 70). Projektgruppen wie *Arch Linux* oder *jEdit* (Texteditor) entsprechen dieser Definition von Peer-Production-Communitys passgenau, allerdings spielen ihre Produkte für den allgemeinen Softwaremarkt kaum eine Rolle und die Zahl der involvierten Entwickler ist überschaubar. Größere Vorhaben wie die *GNU Compiler Collection* oder *KDE* (Desktop-Umgebung) kommen hingegen inzwischen nicht mehr ohne basale Entscheidungshierarchien aus. Selbiges gilt ebenso für *LibreOffice*, das sich als legitime Fortführung von *OpenOffice.org* versteht (s. Tab. 11).

Tabelle 11: Kennzahlen ausgewählter Peer-Production-Communitys

	geschätzter Aufwand 9/2015, Basic COCOMO	Commits (5/2014 bis 5/2015)	Lizenz-modell	strategische Kontrolle
GNU CC	~ 2.200 Personenjahre	8.382	strongly protective	Steuerungskomitee
KDE	~ 7.000 Personenjahre	36.220	(strongly) protective	Core Team
LibreOffice	~ 240 Personenjahre	21.384	weakly protective	Komitee / Board
jEdit	~ 78 Personenjahre	143	strongly protective	(B. Kautler) / Team
Arch Linux	~ 19 Personenjahre	323	(strongly) protective	(A. Griffin) / Team

Quelle: *Open Hub*; eigene Recherchen. * Basic Constructive Cost Model (organisch [a=2.4, b=1.05])

Arch Linux

Arch wurde 2002 als *GNU/Linux*-Distribution für Fortgeschrittene ins Leben gerufen und gilt unter Experten als beliebt, da es seinen Nutzern viele Freiheiten lässt:

> "Installing Arch Linux is a bit like building your own house. You have to dig the foundation, erect the walls, build the roofs, run the plumbing and electrical wiring around it [...]. Installing Arch Linux is not at all like renting an apartment, just moving in, and letting the landlord take care of everything else." (Bhartiya 2015)

Die Entwicklerbasis für das Kernsystem ist mit rund 30 aktiven freiwilligen Programmierern relativ klein; ihre Koordination erfolgt über *Internet Relay Chat*, Foren, Mailinglisten und ein öffentliches Wiki, in dem sich die basalen Richtlinien des Projekts finden. Mit Aaron Griffin verfügt *Arch Linux* über einen nominellen Projektleiter, der aber nicht die Position eines absoluten Entscheiders einnimmt. Entschlüsse werden auf der Grundlage konsensorientierter Diskussionen mit alleinigem Blick auf die beste technische Lösung zwischen den Kernentwicklern gefasst; dezidierte Entscheidungswege und Kriterien für die Aufnahme in diesen inneren Kreis gibt es nicht. Im *Arch Wiki* (2015) findet sich lediglich eine Liste an Vorarbeiten, die Aspiranten dabei helfen können, "to gain some 'popularity' towards Arch's developers". Laut Griffin (in Cunningham 2010) folgt das Vorhaben ausschließlich intrinsischen Motiven: "We don't make Arch so that we can win the most users, or get piles of cash. We make Arch because this is the OS we want to use." *Arch* erhält Kleinspenden von Privatpersonen, aber keine nennenswerte Unterstützung durch die Softwareindustrie.

jEdit

Auch *jEdit* verfügt über keine korporativen Sponsoren und richtet sich als *Java*-basierter Texteditor für Programmierer an einen verhältnismäßig kleinen Nutzerkreis. Das Community-Wiki bietet den Dreh- und Angelpunkt für die Projektkommunikation und listete Mitte 2015 über 300 Nutzer auf, von denen sich laut *Open Hub* (2015) allerdings zwischen 7/2014 und 7/2015 lediglich neun Personen in die zwischenzeitlich für einige Monate ruhende Produktentwicklung eingebracht haben, darunter an vorderster Stelle Björn Kautler (alias ‚Vampire') und Alan Ezust, die gemessen an der Außenkommunikation der Gemeinschaft aus offenkundig nicht persönlich bekannten Entwicklern seit mehreren Jahren als informelle Projektmanager fungieren. Darüber hinaus verfügt die offene Entwicklergemeinschaft ("You can 'join' simply by subscribing to the [...] mailing lists") über keine fixierten Arbeits-

regeln oder Rollenverteilungen; die Koordination findet hauptsächlich über eine projekteigene Web-Plattform inklusive Foren, Dateiverwaltung und Mailinglisten statt. Über korporative Sponsoren verfügt das Vorhaben nicht; es bietet aber Privatpersonen die Möglichkeit, kleinere Beträge zu spenden (Einnahmen 2007–2015: ca. 2000 US-Dollar), über deren Verwendung in der Projektgemeinschaft noch diskutiert wird. *jEdit* wird oft als typisches Beispiel für ein "traditional OSS system" (Capiluppi et al. 2012: 197) in Benklers Sinne genannt; sowohl die Marktrelevanz des Texteditors als auch das Aktivitätsniveau in der Gemeinhaft hat zuletzt allerdings abgenommen.

KDE

Die seit 1996 entwickelte *Linux*-Arbeitsplatzumgebung *KDE* hingegen findet in öffentlichen Bildungs- und Verwaltungseinrichtungen weltweit verbreiteten Einsatz (Schwerpunkt: Lateinamerika) und diente in den letzten Jahren als Ausgangsbasis für zahlreiche Projekte größerer Softwareunternehmen (z. B. *WebKit*). Dementsprechend beteiligen sich viele mittlere und größere Firmen an der Entwicklung der *KDE*-Pakete, darunter *Google*, *SUSE* und *Blue Systems*, welche das Projekt über den gemeinnützigen *KDE e.V.* auch finanziell unterstützen (vgl. Kap. 3.4). Nichtsdestotrotz orientiert sich die Projektgemeinschaft nach wie vor an den originären Maximen freier Softwareentwicklung und bemüht sich um egalitäre Koordinationsweisen. 2014 bestand die *KDE*-Community aus 500 aktiven Entwicklern, die sich über Mailinglisten, Foren und Repositorien entlang von Modulen selbstgesteuert organisieren. Anders als *Debian* verfügt *KDE* über keine fixierte Verfassung, keinen Projektmanager und auch der gewählte Vorstand des *KDE*-Vereins greift nicht in die konkrete Softwareentwicklung ein (vgl. Alleyne 2011). Gleichwohl haben sich aufgrund der hohen Zahl an Entwicklern übergeordnete Managementstrukturen ausgebildet: 2008 wurde eine ‚Community Working Group' installiert, die als ‚mediator upon request' agiert; daneben entscheidet ein mehrere Dutzend Personen umfassendes ‚KDE Core Team' öffentlich und konsensorientiert über die weitere strategische Ausrichtung des Projekts. Diese Gruppe lässt sich aber nicht als geschlossener Führungszirkel verstehen: "[...] membership can be difficult to strictly define at any point in time. Anyone can become a member of this core group, the particular contributor must however have distinguished him/herself through outstanding work and dedication [...]" (KDE 2015).

LibreOffice

LibreOffice weist im Vergleich dazu explizitere Abstufungen in der Projektsteuerung auf. *LibreOffice* ist eine Abspaltung von *OpenOffice.org*, die 2010 durch die Kernentwickler der Community betrieben wurde, da diese mit der Projektpolitik des damals anleitenden Unternehmens *Oracle* unzufrieden waren. Mittlerweile ist *LibreOffice* die empfohlene Bürosoftware aller führenden *Linux*-Distributionen und verfügt über eine deutlich aktivere Projektgemeinschaft als *OpenOffice*, wobei knapp 40 Prozent der Änderungen zwischen 3/2014 und 3/2015 (22.134 Commits) von bei *Red Hat* angestellten Entwicklern getätigt worden sind (vgl. Corbet 2015). Die *LibreOffice*-Gemeinschaft organisiert sich weithin selbstständig entlang offener Arbeitsgruppen über webbasierte Kollaborationswerkzeuge; sie untersteht dabei aber unmittelbar den strategischen Setzungen und dem Qualitätsmanagement der mit dem Projekt gegründeten *Document Foundation*, die über ein gewähltes ‚board of directors', ein meritokratisch bestimmtes Steuerungskomitee und ein ‚advisory board' verfügt. Mitglieder dieses Beirats sind Organisationen, die einen jährlichen Beitrag zwischen 5.000 und 20.000 US-Dollar entrichten oder Mitarbeiter für die Projektentwicklung abstellen, darunter *Google*, *Red Hat*, *SUSE*, *AMD*, *Intel* sowie die Stadt München (Stand: 10/2015). Zudem erhält die *Document Foundation* (Einnahmen 2014: 0,8 Mio. Euro) finanzielle Unterstützung durch ein breites Portfolio an Softwareunternehmen.

GNU Compiler Collection (GCC)

Die *GNU Compiler Collection* (*GCC*), welche seit 1987 als Teil des durch Richard Stallman initiierten *GNU*-Projekts entwickelt wird, verfügt inzwischen ebenso über ausdefinierte Managementstrukturen. Die *GCC* übersetzt den Code höherer Programmiersprachen in Prozessor- bzw. Maschinensprache und findet in zahlreichen informationstechnischen Produkten Einsatz, weshalb unternehmensaffiliierte Programmierer intensiv in das Vorhaben involviert sind. Aus diesem Grund wurde bereits 1998 ein *GCC Steering Committee* installiert – "with the intent of preventing any particular individual, group or organization from getting control over the project" (gcc.gnu.org/steering.html). Dieses konsensorientierte Komitee setzt sich neben Stallman aus 13 Mitgliedern zusammen, welche die verschiedenen Anspruchsgruppen des Vorhabens repräsentieren sollen. Neun Mitglieder dieser Führungsgruppe nehmen diese Rolle seit ihrer Einrichtung 1998 ein; zehn Mitglieder arbeiten bei größeren IT-Unternehmen wie *IBM*, *Google*,

Red Hat und *SUSE*. Zudem verfügt *GCC* über einen Release Manager, einen fixierten Entwicklungsplan, der auf der Diskussion in Mailinglisten basiert, ‚global reviewers' für das Gesamtprojekt und ‚maintainers' für die informell strukturierten Arbeitsmodule. Anders als im Falle des *Linux Kernels* steht an der Spitze dieses Entwicklungsmodells allerdings kein ‚benevolent dictator': "[...] the project recognizes the historical importance of Richard Stallman, but takes decisions in general by consensus" (Gonzalez-Barahona/Robles 2013: 177). Finanzielle Unterstützung erhält das Projekt über die *Free Software Foundation*, die 2013 ca. 1,2 Mio. US-Dollar an Spenden erhielt (u. a. von *Google*, *IBM*, *HP*) und 0,6 Mio. US-Dollar an Personalkosten auswies, die primär in die punktuelle Anstellung von Programmierern für spezifische Entwicklungsarbeiten geflossen sind.

Auch in klassischen Peer-Production-Communitys bilden sich insofern mit wachsender Größe hierarchische Entscheidungsstrukturen mit eindeutigen Machtasymmetrien und definierten Richtlinien heraus, die zur Kohärenz der Projekte beitragen und die übergreifende Koordination erleichtern. Unterhalb dieser formalen Managementstrukturen lässt sich im Unterschied zu elitezentrierten Gemeinschaften indes ein heterogeneres Bouquet an informellen Arbeitsgruppen ausmachen. Im Falle von *GNU CC*, *KDE* und *LibreOffice* bestimmt sich die Projektaktivität heute gleichwohl zu einem großen Teil durch die Beiträge angestellter Entwickler, die sich weniger aus individueller Passion engagieren, sondern weil sie dafür bezahlt werden. Die Entwicklergemeinschaften *Arch* und *jEdit* richten ihre Produkte hingegen auf sehr spezifische Anspruchsgruppen aus, sind für korporative Stakeholder eher uninteressant, werden durch kleine Entwicklerteams getragen und konnten daher auf die Ausbildung ausgeprägter sozialer Strukturierungen verzichten. Sobald allerdings die Gemeinschaft wächst und sich ihre Interaktionen mit externen Akteuren intensivieren, werden augenscheinlich trotz aller technischen Effektivierungen auch in gesellschaftsethisch fundierten Projekten ‚kathedralartige' Koordinationsstrukturen notwendig. Auch ein Basar kann sich nur bis zu einem gewissen Grad selbst organisieren.

5 Bilanz: Open Source als Utopie, Methode und Innovationsstrategie

Unter der Bezeichnung ‚Open-Source-Communitys' wird heute ein breites Spektrum an sehr unterschiedlich ausgerichteten Projekten in der Softwareentwicklung zusammengefasst. Der überwiegende Teil dieser Vorhaben lässt sich freilich kaum mehr mit Eric S. Raymonds (1999: 21) Idee eines selbstorganisierten "great babbling bazaar" oder Yochai Benklers (2002: 375) Vorstellung einer "commons-based peer production" als "phenomenon of large- and medium-scale collaborations among individuals that are organized without markets or managerial hierarchies" in Bezug bringen. Vielmehr haben die meisten marktrelevanten Projektgemeinschaften inzwischen sehr prägnante, in aller Regel hierarchisch abgestufte Führungs- und Entscheidungsstrukturen ausgebildet, folgen übergreifenden Entwicklungsplänen und sind oftmals in einem signifikanten Maße von dem personellen wie finanziellen Involvement größerer Technologieunternehmen abhängig.

In der Retrospektive zeigt sich zudem, dass die quelloffene und proprietäre Entwicklung seit jeher ineinander verwoben sind: Erste Interessengemeinschaften von Computernutzern wurden in den 1950er-Jahren durch Unternehmen mitbegründet; die ab den 1960er-Jahren im akademischen Milieu entstandenen Hackergruppen verwendeten oft gespendetes Equipment; staatliche sowie privatwirtschaftliche Einrichtungen investierten zur gleichen Zeit bereits intensiv in universitäre Softwareprojekte; die Hobbyistenszene der 1970er-Jahre fußte auf ebendiesen Vorarbeiten. Das gemeinsame Problem dieser meist in informellen Zusammenhängen entwickelten Architekturen bestand indes in ihrer mangelnden rechtlichen Absicherung und mitunter raschen Kommodifizierung: Die Ergebnisse der Zusammenarbeit wurden – wie bereits in früheren Prozessen kollektiver Invention (vgl. Allen 1983) – in der Regel als gemeinfreie Produkte veröffentlicht und waren daher nur unzureichend vor Einzelaneignung geschützt. Die eigentliche Geschichte der Open-Source-Softwareentwicklung beginnt vor diesem Hintergrund erst mit den Ende der 1980er-Jahre definierten neuartigen Lizenzmodellen und lässt sich in drei Phasen einteilen (s. Abb. 10).

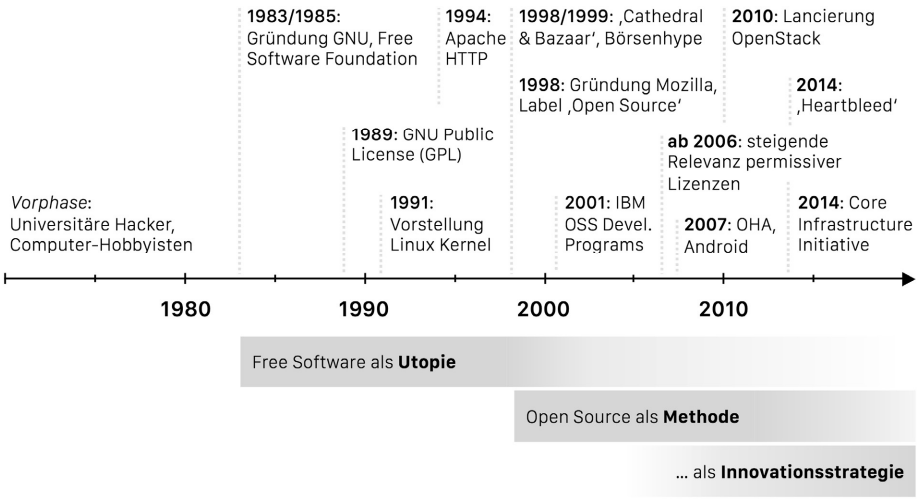

Abb. 10 Phasen der Open-Source-Softwareentwicklung

Ab 1983 wurde ‚Free Software' durch Richard Stallman zunächst als gesellschaftsethisch fundierte Alternative zur proprietären Softwareherstellung und als *utopischer Gegenentwurf* zu kapitalistischen Wirtschaftsstrukturen formatiert. Die durch ihn gegründete *Free Software Foundation* erarbeitete in den Folgejahren die ersten rechtlich belastbaren ‚Copyleft'-Lizenzen, welche garantieren, dass auch Weiterentwicklungen freier Software einzig unter den gleichen Bedingungen verbreitet werden dürfen. Bereits in dieser Phase quelloffener Software als proklamierter Vorbote einer "high-tech gift economy" oder "really existing form of anarcho-communism" (Barbrook 1998; vgl. später ähnlich: Benkler 2002, 2013) wurden allerdings vielversprechende Projekte wie der *Linux Kernel* durch Unternehmen wie *IBM* unterstützt.

Befördert durch den New-Economy-Hype der ausgehenden 1990er-Jahre und die Vermarktungsbemühungen der 1998 gegründeten *Open Source Initiative* avancierte die quelloffene Softwareentwicklung mit der Jahrtausendwende zunehmend zu einer *allgemeinen Branchenmethode*. Kleinere und größere Technologieunternehmen investierten stetig intensiver in Open-Source-Projekte, um für ihre Belange förderliche Infrastrukturen und Standards zu protegieren, die Interoperabilität eigener Lösungen abzusichern oder ihre proprietären Programmpakete mit quelloffenen Komponenten zu ergänzen. Darüber hinaus bildete sich eine Vielzahl an Start-up-Firmen heraus, die mit quelloffener wie kostenfreier Software im Verbund mit kostenpflichtigen Beratungs- und Supportdienstleistungen ein Geschäft aufbauen wollten.

Parallel dazu ist ‚Open Source' seit Mitte des letzten Jahrzehnts zu einem *festen Bestandteil der Innovationsstrategien* aller etablierten Anbieter geworden, welche so ihre internen Forschungs- und Entwicklungsaktivitäten, die auf dem volatilen Markt für Kommunikations- und Informationstechnologien im Normalfall durch strikte Schließung und Geheimhaltung geprägt sind, um lizenzrechtlich abgesicherte Kollaborationskontexte mit anderen Marktteilnehmern und kreative Spielflächen ergänzen. *Apple* etwa hat neben *WebKit* jüngst die Programmiersprache *Swift* unter freie Lizenz gestellt; *Google* hat sich bereits 2007 dazu entschlossen, *Android* als Open-Source-Projekt anzulegen. Diese Entscheidungen resultieren allerdings weniger aus ideeller Verbundenheit mit quelloffener Software, sondern aus ökonomischem Kalkül – zum Beispiel, um die Verbreitung eigener Softwarearchitekturen zu erhöhen, um herstellerspezifische Hardware- und Softwareumgebungen für Entwickler interessant zu machen oder um neue Geschäftsfelder zu erschließen.

In den zurückliegenden 20 Jahren ist die Open-Source-Entwicklung auf diese Weise zunehmend zu einem integralen Bestandteil der Softwareindustrie geworden; sie hat dabei aber ihre Formatierung als Gegenentwurf zur kommerziellen Herstellung weitgehend verloren – auch wenn sie in sozialwissenschaftlichen Beschreibungen nach wie vor immer wieder als pauschales Beispiel für eine marktunabhängige kollektive Produktion genannt wird (vgl. zuletzt z. B. in Bennett et al. 2014). Derzeit lassen sich vier idealtypische Varianten von Open-Source-Projekten unterscheiden, die sich mit Blick auf ihre vorherrschenden Koordinationsmuster und dem Grad ihrer Nähe zu etablierten Unternehmen wie folgt voneinander abheben (s. Tab. 12):

- *Korporativ geführte Kollaborationsprojekte* (z. B. *Android*, *WebKit*) sind durch Einzelunternehmen oder Unternehmensverbünde gegründet worden, eindeutig an deren Interessen orientiert und zeichnen sich durch entsprechend deutliche Hierarchisierungen auf Arbeitsebene aus. Ihre Teilnehmerbasis speist sich vorrangig aus den Mitarbeiterkreisen der involvierten Firmen, die diese quelloffenen Projektzusammenhänge nutzen, um außerhalb formal gefasster Kooperationsbeziehungen punktuell mit anderen industriellen Stakeholdern (oft auch Konkurrenten) zu kollaborieren und dabei häufig marktzentrale Produkte zu erarbeiten.
- *Heterarchisch angelegte Infrastrukturvorhaben* (z. B. *Eclipse*, *Apache HTTP Server*) sind in der Regel graduell gewachsen und werden nicht durch Unternehmen angeleitet, fußen heute in ihrer Operationsfähigkeit allerdings wesentlich auf dem finanziellen wie personellen Engagement mittlerer und großer IT-Firmen. Sie verfügen über stabile eigenständige

Koordinationsstrukturen, weisen wechselnde Beteiligungsmuster auf, sind in ihrer Grundanlage eher horizontal strukturiert, verteilen Funktionsrollen üblicherweise meritokratisch und arbeiten oft unter der strategischen Führung einer assoziierten Dachorganisation bzw. Stiftung.

- *Elitezentrierte Projektgemeinschaften* (z. B. *Linux Kernel*, *Ubuntu*) stehen ebenfalls nicht unter der direkten Kontrolle eines gewerblichen Akteurs; auch sie stützen sich aber inzwischen zu einem Gutteil auf die Beiträge unternehmensaffiliierter Entwickler. Ihre Koordination erfolgt entlang selektiv-hierarchischer Entscheidungspyramiden, an deren Spitze meist entweder ein langfristig installiertes Führungsteam oder der Projektgründer als ‚benevolent dictator' steht, der mitunter in einem erheblichen Umfang an der Finanzierung des Projekts beteiligt ist.
- *Peer-Production-Communitys* (z. B. *Arch Linux*, *jEdit*) dienen ihrem eigenen Anspruch nach der marktunabhängigen, gleichberechtigten und selbstgesteuerten Kollaboration unter freiwilligen Entwicklern; sie bilden jedoch – wie sich am Beispiel der *GNU Compiler Collection* zeigen lässt – ab einer gewissen Marktrelevanz und Größe der Projektgemeinschaft ebenfalls herausgehobene Führungsstrukturen und basale hierarchische Abstufungen aus, welche die übergreifende Koordination erleichtern und zur Kohärenz der Entwicklungsvorhaben beitragen.

Tabelle 12: Open-Source-Softwareprojekte – idealtypische Ausprägungen

	korporativ geführte Kollaborationsprojekte (z. B. *Android*, *WebKit*)	**heterarchisch angelegte Infrastrukturvorhaben** (z. B. *Apache HTTP Server*)	**elitezentrierte Projektgemeinschaften** (z. B. *Linux Kernel*, *Ubuntu*)	**Peer-Production-Communitys** (z. B. *Arch Linux*)
strategische Führung	Einzelunternehmen / Unternehmensgruppe	Vorstand der Dachstiftung / -organisation	Projektgründer / langfristige Projektleitung	Komitee / Kernteam
Finanzierung	beteiligte Unternehmen	primär Zuwendungen von Unternehmen	korporative Spenden / gemischte Quellen	primär private Kleinspenden
Teilnehmerbasis	Mitarbeiter aus den beteiligten Unternehmen	angestellte Entwickler / explizite Unternehmensvertreter	angestellte und freiwillige Entwickler	freiwillige Entwickler
Arbeitsorganisation	primär hierarchisch	horizontal – meritokratisch	hierarchisch – autokratisch	primär egalitär
Marktrelevanz	hoch	hoch	mittel / hoch	niedrig / mittel

Der gemeinsame Nenner aller betrachteten Entwicklungsvorhaben besteht in den dahinterliegenden quelloffenen Lizenzmodellen, die ihre Produkte wirksam vor direkter Proprietarisierung schützen und einen erwartungssicheren institutionellen Rahmen für die projektbezogene Zusammenarbeit zwischen Einzelentwicklern sowie Unternehmen bieten. Mit „Rebel Code“ (Moody 2002) hat all dies gleichwohl nicht mehr viel gemein: Die Verschränkungen mit marktlichen Kontexten sind in vielen Fällen ausgeprägt, offener Quellcode mündet offenkundig nicht unmittelbar in transparenteren Koordinationsmustern als in anderen Arbeitszusammenhängen und trotz der technisch erweiterten Austauschmöglichkeiten bilden sich mit zunehmender Größe der Projektgemeinschaften regelmäßig hierarchisch gegliederte Entscheidungsstrukturen heraus. Zwar finden modulare und iterative Methoden auf operativer Ebene in allen diskutierten Vorhaben – wie seit geraumer Zeit auch in der professionellen Softwareentwicklung (vgl. z. B. Cerri/Fuggetta 2007; Fuggetta 2003) – verbreitete Anwendung; welchen langfristigen Schwerpunkten das Projekt folgt, definieren aber im Regelfall jeweils nur wenige Entscheidungsträger.

Entgegen dem Eindruck, “that the world of work is changing and that organizations (corporations, not-for-profits, universities) really don’t matter as much as they used to” (Suddaby 2013: 1009), verlieren korporative Akteure in der Open-Source-Softwareprojekten überdies keineswegs an Bedeutung, sondern bleiben als deren Initiatoren, Financiers und Arbeitgeber der involvierten Programmierer prominent im Spiel. Privatwirtschaftliche Unternehmen, staatliche und wissenschaftliche Organisationen können ihre Ressourcen im Normalfall systematischer und auf lange Sicht deutlich verlässlicher als individuelle Beiträger (z. B. Studierende, Hobbyisten, Freizeitprogrammierer) in die Projektgemeinschaften einbringen, tragen so zu einer Erhöhung der Planungssicherheit in den jeweiligen Kontexten bei und verfügen mithin oft über einen nicht zu unterschätzenden Einfluss auf deren Anlage und Orientierung. Ferner verfügen unabhängige Projekte, die nicht unter der direkten Ägide eines Unternehmens stehen, zumeist über assoziierte gemeinnützige Organisationen, die als adressierbare Dachidentitäten für externe Stakeholder dienen, die Ausrichtung des jeweiligen Vorhabens mitbestimmen und die Gemeinschaft bei Konflikten oder auseinanderstrebenden Partikularinteressen stabilisieren können (vgl. Ahrne et al. 2015; Ahrne/Brunsson 2011; klassisch zu Organisationen: March/Simon 1958; Blau/Scott 1962).

Das nach wie vor einschlägige Narrativ von der quelloffenen Softwareentwicklung als revolutionäres Produktionsmodell, das auf freiwilliger sowie

selbstgesteuerter Kollaboration beruht, eingespielten sozioökonomischen Koordinationsweisen wie ‚Markt' und ‚Hierarchie' überlegen ist und etablierte Unternehmen zu weitreichenden Anpassungen zwingt, lässt sich in dieser Radikalität insofern in den allgemeinen Strom der übersteigerten Entdifferenzierungserwartungen der digitalen Moderne einordnen (vgl. Dickel/Schrape 2015): Ebenso wie im Falle anderer Spielarten kollektiven Handelns (z. B. in sozialen Bewegungen) oder kollektiver Produktion (z. B. in Wikis) oder mit Blick auf den fortlaufenden Strukturwandel der Öffentlichkeit zeigt sich auch hinsichtlich Open-Source-Communitys, dass die Online-Technologien zwar die Kommunikation und Koordination erheblich effektivieren und neue Möglichkeiten der Zusammenarbeit aufgeschlossen haben, aber grundlegende gesellschaftliche Strukturierungen und Rollendifferenzierungen dadurch keineswegs obsolet werden (vgl. Dolata/Schrape 2015; Schrape 2012). Die meisten der hier diskutierten Projektgemeinschaften haben sich mittlerweile denn auch zu äußerst komplexen sozialen Gebilden entwickelt, in denen sich klassische Organisationsprinzipien mit selektiv-hierarchischen Entscheidungsmustern und Einflussasymmetrien widerspiegeln.

Technische Infrastrukturen können die ortsunabhängige Kollaboration vereinfachen und die konkrete Arbeitsorganisation unterstützen; die sozialen Ordnungsleistungen, die einen volatilen interessenbasierten Zusammenschluss in eine festgefügte Projektgemeinschaft mit der Fähigkeit zu strategischem Handeln überführen, können sie aber nicht ersetzen. Insoweit waren ab den 1980er-Jahren nicht nur die neuen Formen der elektronischen Vernetzung, sondern ebenso die Kristallisation übergreifend akzeptierter Werte und Arbeitskonventionen sowie vor allem anderen die Definition rechtlich tragfähiger Lizenzmodelle für die Ausdifferenzierung und den anhaltenden Erfolg quelloffener Entwicklungsvorhaben von herausgehobener Bedeutung. ‚Copyleft'-Lizenzen und ihre Derivate, welche eine grundsätzliche Quelloffenheit auch für Weiterentwicklungen freier Software garantieren und so ihre direkte Proprietarisierung verhindern, haben den institutionellen Rahmen für eine auf Dauer gestellte Form kollektiver Invention und Kollaboration aufgespannt, die in den 1980er- und 1990er-Jahren zunächst in geschützten subversiven Nischen Anwendung fand, ab der Jahrtausendwende zunehmend korporative Aneignung erfahren hat und heute in den allgemeinen Kanon der Softwarebranche übergegangen ist. Quelloffene Lizenzen sind insofern nicht einfach nur eine vorübergehende "form of institutional jiujitsu" (Benkler 2002: 446) vor einer bestenfalls erhofften allgemeinen Auflösung geistiger Eigentumsrechte im Softwarebereich, sondern nach wie vor das soziostruktu-

relle Alleinstellungsmerkmal und die elementare Geschäftsgrundlage von Open-Source-Projekten, die inzwischen in einem unauflösbaren Verflechtungszusammenhang mit dem kommerziellen IT-Markt stehen.

Glossar

Android: Quelloffenes Betriebssystem (basierend auf dem *Linux Kernel*) für Smartphones, Mediaplayer und Tablet-Computer, das seit 2007 von der durch *Google* gegründeten *Open Handset Alliance* entwickelt wird und 2015 im Bereich der Mobile Devices einen globalen Marktanteil von weit über 80 Prozent eingenommen hat.

Applikation: Anwendungssoftware, die eine für den Computernutzer nützliche oder gewünschte Funktion leistet (z. B. Textverarbeitung, Bild- und Videobearbeitung, Tabellenkalkulation, Finanzbuchhaltung, Computerspiele, E-Mail-Versand).

Apache HTTP Server: Quelloffene und seit 1996 global meistgenutzte Webserver-Software, die auf Nutzeranfrage (z. B. durch einen Mausklick auf einen Link) über das Internet Dokumente oder Daten zur Verfügung stellt.

Application Programming Interface (API): Schnittstelle zur Anwendungsprogrammierung, die von einem Softwaresystem (z. B. ein Betriebssystem, eine Datenbank oder eine Website) anderen Programmen zur Verfügung gestellt wird, damit diese auf ausgewählte Funktionen oder Daten zugreifen können.

Betriebssystem: Sammlung von Programmen, welche die Hardware-Ressourcen eines Computers oder mobilen Geräts verwalten und der Anwendungssoftware des Benutzers zur Verfügung stellen. Ein Betriebssystem besteht im Regelfall aus einem Kernel, der direkten Zugriff auf die Hardware des Computers hat und die basale Prozess- und Datenorganisation definiert, sowie spezifischeren Programmen, die beim Start des Systems bedarfsbezogen geladen werden (z. B. Gerätetreiber).

Big Data: Große Datenmengen aus einer Vielzahl unterschiedlicher Quellen, die z. B. durch die Kommunikation im Web (Social-Networking-Dienste, App Stores etc.) generiert werden und mithilfe neuartiger Auswertungstechnologien synthetisiert, durchsucht, analysiert und visualisiert werden können.

Blog: Kurzform für Weblog, ein Kofferwort aus ‚Web' und ‚Log'. Überbegriff für tagebuchartig geführte, in der Regel chronologisch sortierte und mehr oder minder regelmäßig durch ein oder mehrere Personen aktualisierte Inhaltsangebote auf einer Website, die oft durch die Leser kommentiert werden können.

Cloud: Serverarchitekturen, welche das dezentrale Speichern von Daten und Inhalten im Netz ermöglichen, die danach von einem Computer oder Mobilgerät mit Online-Zugriff abgerufen werden können.

Code: Programmanweisungen, welche die Funktionalität einer Software in einer bestimmten Programmiersprache beschreiben. Die für den Menschen lesbare und bearbeitbare Form von Programmcode wird als ‚Source-Code' bezeichnet.

Commons-based Peer Production: Ein von Yochai Benkler ab 2002 am Beispiel der Open-Source-Softwareentwicklung explizierter neuartiger Produktionsmodus, der als Alternative zu eingespielten ökonomischen Koordinationsweisen wie ‚Markt' und ‚Hierarchie' gefasst wird und eine selbstgesteuerte, gleichberechtigte wie freiwillige Zusammenarbeit einer Vielzahl an Menschen im Online-Kontext beschreibt.

Community of Interest: Gruppe bzw. Gemeinschaft, die sich durch bewusst geteilte sowie an spezifischen Interessen ausgerichtete Zielsetzungen, Grundsätze und Wirklichkeitssichten auszeichnet.

Compiler: Software, die den Source-Code eines Computerprogramms in eine maschinenlesbare Sprache übersetzt.

Content-Management-System: (Heute zumeist datenbankbasierte) Software zur Verwaltung, Organisation und (gemeinschaftlichen) Bearbeitung von Inhalten (oft auf Websites).

Copyleft: Eine ursprünglich im Kontext der freien Softwareentwicklung definierte Klausel in Nutzungslizenzen, die festlegt, dass Weiterentwicklungen eines Werkes mit den gleichen originären Freiheiten weitergegeben werden müssen und nicht mit eigenen Restriktionen belegt werden dürfen. Die erste ausformulierte ‚Copyleft'-Lizenz war 1989 die *GNU General Public License*.

Copyright: Urheberrechtsform (USA, Großbritannien), die definiert, wer ein Werk verwerten bzw. vervielfältigen darf und durch den Urheber vollständig auf den Verwerter übertragen werden kann. Das Copyright erlischt in der Regel 70 Jahre nach dem Tod des personellen Urhebers, wodurch das Werk Allgemeingut wird.

Creative Commons: 2001 gegründete gemeinnützige Organisation, die Standard-Lizenzverträge erarbeitet, mit denen ein Autor der Öffentlichkeit unterschiedliche Nutzungsrechte an seinen urheberechtlich geschützten Werken einräumen kann. Einige Lizenzen schränken die Nutzung ein; andere sorgen dafür, dass auf einen Schutz so weit wie möglich verzichtet wird.

Crowd: Abgrenzbare Menge an Onlinern, die sich spontan bzw. situativ entlang benennbarer Themenstellungen oder Ereignisse ausrichtet, aber (noch) keine verfestigten Gruppenstrukturen ausgebildet hat.

Crowdfunding: Mit dem Social Web populär gewordene Finanzierungsform für Projekte verschiedenster Art (z. B. Bücher, Musik, IT-Produkte, Geschäftsideen). Über Crowdfunding-Plattformen und virale Aufmerksamkeitskampagnen wird angestrebt, möglichst viele Onliner dazu zu bewegen, freiwillig (kleine) finanzielle Beträge in die präsentierte Idee zu investieren.

Crowdsourcing: Auslagerung eigentlich unternehmensinterner Aufgaben an eine offene Gruppe freiwilliger und oft unbekannter User, die meistens für ihre Arbeiten nicht oder nur unzureichend entlohnt werden (z. B. in Beta-Programmen).

Digital Natives: Bezeichnung für Personen, die mit dem Computer und dem Internet aufgewachsen sind und sich aus diesem Grund mit diesen Techniken besser auskennen sollen als ‚Digital Immigrants', welche erst im Erwachsenenalter mit den neuen Kommunikations- und Informationsmöglichkeiten in Berührung gekommen sind.

Distribution: Sammlung von Software, die als Komplettpaket weitergegeben wird. Eine *Linux*-Distribution besteht z. B. aus einem Betriebssystem-Kern (dem *Linux Kernel*), Gerätetreibern, einer grafischen Benutzeroberfläche, Programmbibliotheken (oft *GNU*-Komponenten), Anwendungssoftware und Dokumentationsdateien.

Eclipse: 2001 von *IBM* freigegebenes quelloffenes Programmierwerkzeug, das ursprünglich als integrierte Entwicklungsumgebung für die Programmiersprache *Java* angelegt war, aber mittlerweile auch für viele andere Aufgaben eingesetzt wird.

F(L)OSS: Akronym für ‚Free/Libre Open Source Software' bzw. ‚Free and Open Source Software', das in der sozialwissenschaftlichen Literatur und der öffentlichen Kommunikation genutzt wird, um den seit Ende der 1990er-Jahre gärenden Namens- und Ausrichtungsstreit zwischen der gesellschaftspolitisch fundierten *Free Software Foundation* und der pragmatisch orientierten *Open Source Initiative* zu umgehen.

Fork: Abspaltung eines neuen Projekts in der Softwareentwicklung; Entwicklungszweig nach der Aufspaltung eines Projektes in mehrere Folgeprojekte, in denen der Quelltext unabhängig vom ursprünglichen Mutterprojekt weiterentwickelt wird.

Free Software Foundation: 1985 von Richard Stallman gegründete Stiftung, die der Förderung freier Softwareentwicklung dient.

Free Software: Der Begriff wurde ab 1983 durch Richard Stallman geprägt, der mit *GNU* das erste freie Softwareprojekt initiierte. Im Gegensatz zu proprietärer Software zeichnet sich freie Software dadurch aus, dass ein Nutzer mit Empfang der Software sämtliche Nutzungsrechte mitempfängt, d. h. die Software unbegrenzt verwenden darf, die Software kopieren und weitergeben kann sowie die Freiheit besitzt, die Software weiterzuentwickeln und in veränderter Form zu veröffentlichen.

GNU: Rekursives Akronym für „GNU is not Linux“; 1983 durch Richard Stallman initiiertes Kollaborationsprojekt mit dem Ziel, ein vollkommen freies unixoides Betriebssystem zu entwickeln. Der *GNU*-Kernel ist bis heute nicht für den praktischen Einsatz geeignet; allerdings werden funktionsfähige *GNU*-Pakete wie die *GNU Compiler Collection* und *GNU* Libraries regelmäßig mit dem *Linux Kernel* kombiniert, was zu der bis heute ausgetragenen *GNU/Linux*-Namenskontroverse geführt hat.

Internet: Sammelbegriff für ein globales Computer-Netzwerk. Im technischen Sinne bezeichnet das Internet ein weltumspannendes Netzwerk über das Transmission Control Protocol/Internet Protocol (TCP/IP) miteinander verbundener Computer. Auf dieser Grundlage werden viele verschiedene Dienste, wie z. B. E-Mail, das *World Wide Web* oder das *Usenet* angeboten.

Kernel: Kern und zentraler Bestandteil eines Betriebssystems mit direktem Zugriff auf die Hardware, der die grundsätzliche Prozess- und Datenorganisation festgelegt, auf der alle weiteren Softwarebestandteile eines Computersystems aufbauen.

LibreOffice: Freie Zusammenstellung gebräuchlicher Bürosoftware, die in ihrem Funktionsumfang vergleichbar mit *Microsoft Office* ist; 2010 aus *OpenOffice.org* hervorgegangene Abspaltung (Fork), die unabhängig weiterentwickelt wird und mittlerweile das empfohlene Office-Paket in gängigen *Linux*-Distributionen ist.

Linux Kernel: 1991 durch Linus Torvalds veröffentlichter *Unix*-ähnlicher Kernel, der 1992 unter die *GNU General Public License* gestellt wurde und heute die Basis für viele Betriebssysteme bildet (u. a. auch *Android*). Im allgemeinen Sprachgebrauch bezeichnet *Linux* nicht nur den Kernel selbst, sondern ganze *Linux*-basierte Systeme, die den *Linux Kernel* mit einer Vielzahl anderer Komponenten kombinieren.

Lizenz: Allgemein ‚Erlaubnis‘; bei urheberrechtlich geschützten Werken (z. B. Software) durch den Urheber über einen Lizenzvertrag eingeräumte Rechte, das Werk auf eine bestimmte Weise bzw. unter festgelegten Bedingungen zu nutzen.

Middleware: Programmarchitekturen, die zwischen Betriebssystem und Anwendungen vermitteln und einen wechselseitigen Datenaustausch ermöglichen.

Mozilla Firefox: Ab 2002 unter dem Namen *Phoenix* entwickelter Webbrowser; (zunächst experimentelle) Abspaltung der auf dem 1998 freigegebenen Source-Code des *Netscape Communicators* basierenden *Mozilla Application Suite*; 2004 unter dem Namen *Firefox* in erster Version veröffentlicht. Zwischen 2007 und 2011 war *Mozilla Firefox* nach dem *Microsoft Internet Explorer* der global meistgenutzte Browser; 2015 lag sein weitweiter Marktanteil bei 15 bis 20 Prozent.

OpenOffice: 2000 auf der Grundlage des offengelegten Quellcodes von *StarOffice* initiiertes und seit 2011 unter Federführung der *Apache Software Foundation* (davor *Sun*, *Oracle*) entwickeltes Programmpaket für gängige Bürozwecke. Seit Mitte 2014 ist das Projekt weitgehend inaktiv; Basis für das Ende 2010 abgespaltete *LibreOffice*.

Open Innovation: Wirtschaftswissenschaftlich geprägter Begriff, der die Öffnung von vormals unternehmensinternen Innovationsprozessen (‚Closed Innovation') nach außen beschreibt. Durch die strategische Nutzung von Austauschprozessen mit externen Stakeholdern soll sich das Innovationspotenzial einer Organisation erhöhen.

Open Source: Quelloffene Werke (oft Software), die qua ihrer Lizenz nicht nur die Nutzung des Endprodukts, sondern auch die Weiterverarbeitung ihres Quelltextes erlauben. Als allgemeiner Begriff eingeführt wurde ‚Open Source' mit der Gründung der *Open Source Initiative* im Frühjahr 1998.

Open Source Initiative: 1998 von Eric S. Raymond, Bruce Perens, Tim O'Reilly und anderen Szeneprotagonisten gegründete nichtkommerzielle Organisation, welche die *Open Source Definition* aktualisiert und Softwarelizenzen zertifiziert, die den Kriterien dieser Definition entsprechen.

OpenSSL: Weltweit eingesetzte Kryptografie-Software, die für viele verschlüsselten Internet-Verbindungen verwendet und seit 1998 als Open-Source- Projekt entwickelt wird. Erst seit 2014 wird *OpenSSL* durch die Softwareindustrie über die *Core Infrastructure Initiative* in einem signifikanten Maße unterstützt.

OpenStack: Seit 2010 u.a. von der *NASA* sowie den Unternehmen *HP*, *IBM*, *Intel*, *Red Hat*, *Canonical* und *Rackspace* entwickelte freie Cloud-Computing-Architektur.

Open Standards: Technische Standards, Formate und Protokolle, die keinem exklusiven Patent- oder Urheberrechtsschutz unterliegen und öffentlich do-

kumentiert sind (im Bereich des Internets z. B.: TCP/IP, FTP, HTTP, HTML, CSS).

Patch: Codezeilen, die Fehler oder Funktionslücken in einem Programm provisorisch ‚flicken', bis diese in der nächsten Version abschließend behoben werden.

Permissive Lizenz: Open-Source-Lizenz, die eine sehr viel breitere Wiederverwertung erlaubt als strenge Copyleft-Lizenzen. Derivate entsprechend lizensierter Software müssen nicht unter derselben Lizenz veröffentlicht werden wie der Ursprungscode, sondern dürfen z. B. wieder zu proprietärer Software verarbeitet werden.

Proprietäre Software: Software, deren Quellcode im Normalfall nicht veröffentlicht wird, deren Urheber (in der Regel Unternehmen) sich exklusive Rechte zur Veränderung und Verbreitung vorbehalten und Lizenzen für die Nutzung der Programme verkaufen (wie z. B. im Falle von *Microsoft Windows*). Ferner Dateiformate, Standards und Protokolle, die nicht offen einsehbar sind.

Prosumer: 1980 durch Alvin Toffler eingeführter Begriff, um Personen zu beschreiben, die gleichzeitig Konsumenten und (bis zu einem gewissen Grad) auch Hersteller der von ihnen verwendeten Produkte sind, etwa durch die Übernahme von vormals anbieterseitigen Aufgaben oder durch Feedbackleistungen.

Produser: 2006 durch Axel Bruns eingeführter Begriff, um aktive Onliner zu beschreiben, welche sich in Open-Content- oder Open-Source-Projekten – von klassischen Marktkontexten abgekoppelt – an der kollaborativen Weiterentwicklung von Inhalten oder Software beteiligen (z. B. *Wikipedia*-Autoren).

Social Web: Von Howard Rheingold ab 1996 geprägter Begriff, der den nutzerseitigen Austausch im Web im Sinne eines ‚many-to-many'-Mediums betont. Oft als Synonym zu ‚Social Media' oder ‚Web 2.0' verwendet.

Softwareentwickler: Überbegriff für Personen, die an der Erstellung einer Software in verschiedenen Rollen mitwirken: Programmierer sind für die Implementierung der Software zuständig; Softwarearchitekten entwerfen den Aufbau von Softwaresystemen und betrachten dieses von einer abstrakteren Ebene als reine Programmierer; Tester prüfen die Programme auf Fehler; ‚Requirement Engineers' führen Anforderungsanalysen durch und konzipieren entsprechende Anpassungen der Software.

Source-Code: Der für Menschen lesbare, in einer bestimmten Programmiersprache geschriebene Text einer Software. Bevor ein Programm von einem Computer ausgeführt werden kann, muss der Source-Code von einem ‚Compiler'

oder ‚Interpreter' in eine maschinenverständliche Binärdatei übersetzt werden. Solche kompilierten Dateien sind für Menschen nicht mehr lesbar und daher auch nicht veränderbar.

Support: Lösungsorientierte Beratungstätigkeiten (oft für Benutzer von Computern), heute übliches Synonym für allgemeine Kundenbetreuung. Im Softwarebereich u. a. das regelmäßige Bereitstellen von ‚Patches', mit denen Fehler in einem Programm behoben werden, Hilfe und Unterstützung über Telefon-Hotlines sowie das Bereitstellen von Online-Foren für den Austausch von Produktnutzern.

Ubuntu: Eine verbreitete und für Laien leicht zu bedienende Distribution des quelloffenen Betriebssystems *GNU/Linux*, die auf der ebenfalls freien Distribution *Debian* basiert und 2004 durch den Dotcom-Millionär Marc Shuttleworth gegründet wurde.

Unix: Bis Anfang der 1980er-Jahre freies, im universitären Kontext entwickeltes, zwischen 1983 und 2005 primär proprietär lizensiertes Mehrbenutzer-Betriebssystem, welches ursprünglich für professionell eingesetzte Computer (Server, Workstations etc.) konzipiert wurde und heute die Grundlage für zahlreiche verbreitete Betriebssysteme bietet, darunter *Linux*, *Android*, *Apple OS X* und *Apple iOS*.

Urheberrecht: Recht auf den Schutz geistigen Eigentums, das die Interessen eines Urhebers an seinen kreativen Werken schützt, falls diese als originelle und eigenständige Leistung eingestuft werden. Das deutsche Urheberrecht sieht geistiges Eigentum als unveräußerlichen Teil der Person des Urhebers an, der die Nutzung seiner Werke verhindern kann, falls er seine Werkintegrität oder Person bedroht sieht, selbst wenn er alle Nutzungsrechte an andere Stellen (z. B. ein Unternehmen) abgegeben hat. Das US-amerikanische Copyright regelt hingegen primär die ökonomischen Verwertungsrechte an einem Werk.

Usenet: Ab Anfang der 1980er-Jahre in der Computerszene populäres elektronisches Netzwerk, das einen eigenständigen Zweig des Internets darstellt und entlang von Diskussionsforen strukturiert ist, an denen jeder Onliner teilnehmen kann. Das *Binary Usenet* erlaubt zudem auch das Verteilen von Binärdateien (z. B. Musik, Filme).

Web 2.0: 1999 erstmals erwähnter Begriff, der ab 2004 als diskursiver Bezugspunkt für eine veränderte Nutzung und Wahrnehmung des Internets an Popularität gewann (‚Social Media'). Ursprünglich wurde das Schlagwort ‚Web 2.0' primär internetökonomisch belegt und beschrieb neue Geschäftslogiken im Online-Bereich.

WebKit: Quelloffen entwickelte Rendering-Engine für HTML-Inhalte zur Darstellung von Webseiten in Webbrowsern; eine seit 2003 unter der Federführung des Unternehmens *Apple* weiterentwickelte Abspaltung der von der *KDE*-Community entwickelten *KHTML*-Engine; u. a. Grundlage für den Webbrowser *Apple Safari*.

Webserver: Ein mit dem Internet oder dem Intranet verbundener Computer (im engeren Sinne: eine Software auf diesem Computer), der Websites und Daten auf Anforderung bereitstellt, z. B. durch eine Mausklick in einem Browser. Die primäre Aufgabe eines Webservers ist die Auslieferung von statischen Dateien (z. B. Bilder) oder dynamisch erzeugten Inhalten (z. B. nutzerprofilabhängig generierte Websites).

Wiki: Hypertext-System für Onlineinhalte, die nutzerseitig via Browser nicht nur gelesen, sondern auch verändert werden können. Das bekannteste Wiki-Projekt ist *Wikipedia*; die erste verfügbare Wiki-Software *WikiWikiWeb* wurde 1995 freigegeben.

World Wide Web: Via Webbrowser über das Internet abrufbares System elektronischer Hypertext-Dokumente, die durch Hyperlinks verknüpft werden können. 1989 unter dem Namen *Mesh* erstmals in einem Projektantrag skizziert und 1991 durch Tim Berners-Lee im *Usenet* veröffentlicht. 1993 wurde der Quellcode zur allgemeinen Nutzung freigegeben.

Literaturverzeichnis

Ahrne, Göran; Aspers, Patrik; Brunsson, Nils (2015): The Organization of Markets. In: *Organization Studies* 36 (1): 7–27.

Ahrne, Göran; Brunsson, Nils (2011): Organization Outside Organizations: The Significance of Partial Organization. In: *Organization* 18 (1): 83–104.

Allen, Robert C. (1983): Collective Invention. In: *Journal of Economic Behavior and Organization* 4 (1): 1–24.

Alleyne, Brian (2011): Challenging Code: A Sociological Reading of the KDE Free Software Project. In: *Sociology* 45 (3): 496–511.

Amazon Inc. (2015): Form 10-K 2014. Washington: US Securities and Exchange Commission.

Ante, Spencer (2014): Red Hat Plays Hardball on OpenStack Software. In: *The Wall Street Journal* vom 13.5.2014. http://on.wsj.com/14qBpus (11/2015)

Apache Software Foundation (2015a): Bylaws of The Apache Software Foundation. http://www.apache.org/foundation/bylaws.html (11/2015)

Apache Software Foundation (2015b): Form 990. Fiscal Year 2013/2014. http://www.apache.org/foundation/records/990-2013.pdf (11/2015)

Aponovich, David; Powers, Stephen; Kesler, Steven (2014): From Geek To Enterprise Chic. Open Source Digital Experience Tools Gain Traction. Research Report. Cambridge: Forrester Inc.

Arch Wiki (2015): Getting Involved. https://wiki.archlinux.org/index.php/Getting_involved (11/2015)

Arrow, Kenneth J. (1962): Economic Welfare and the Allocation of Resources for Invention. In: Nelson, Richard (Hrsg.): *The Rate and Direction of Inventive Activity. Economic and Social Factors*. Princeton: Princeton University Press, S. 609–626.

Baldwin, Carliss; Hippel, Eric von (2011): Modeling a Paradigm Shift: From Producer Innovation to User and Open Collaborative Innovation. In: *Organization Science* 22 (6): 1399–1417.

Barbrook, Richard (1998): The High-Tech Gift Economy. In: *First Monday* 3 (12). http://firstmonday.org/ojs/index.php/fm/article/view/631/552 (11/2015)

Bashe, Charles J.; Johnson, Lyle R.; Palmer, John H.; Pugh, Emerson W. (1986): *IBM's Early Computers*. Cambridge: MIT Press.

Benkler, Yochai (2002): Coase's Penguin, or, Linux and 'The Nature of the Firm'. In: *Yale Law Journal* 112: 369–446.

Benkler, Yochai (2004): Intellectual Property: Commons-based Strategies and the Problems of Patents. In: *Science* 305 (5687): 1110 f.

Benkler, Yochai (2006): *The Wealth of Networks. How Social Production Transforms Markets and Freedom*. New Haven: Yale University Press.

Benkler, Yochai (2013): Practical Anarchism, Peer Mutualism, Market Power, and the Fallible State. In: *Politics & Society* 41 (2): 213–251.

Benkler, Yochai; Nissenbaum, Helen (2006): Commons-based Peer Production and Virtue. In: *Journal of Political Philosophy* 14 (4): 394–419.

Bennett, W. Lance; Segerberg, Alexandra; Walker, Shawn (2014): Organization in the Crowd: Peer Production in Large-scale Networked Protests. In: *Information, Communication & Societ*y 17 (2): 232–260.

Bergquist, Magnus; Ljungberg, Jan; Rolandsson, Bertil (2012): Justifying the Value of Open Source. In: *ECIS 2012 Proceedings*.
http://aisel.aisnet.org/ecis2012/122/ (11/2015)

Bezroukov, Nikolai (1999a): A Second Look at the Cathedral and the Bazaar. In: *First Monday* 4 (12). http://firstmonday.org/article/view/708/618 (11/2015)

Bezroukov, Nikolai (1999b): Open Source Software Development as a Special Type of Academic Research. In: *First Monday* 4 (10).
http://journals.uic.edu/ojs/index.php/fm/article/view/696/606 (11/2015)

Bhartiya, Swapnil (2015): Five Reasons I Roll with Arch Linux, and Why You Should Too. In: *IT World* vom 18.5.2015.
http://www.itworld.com/article/2898189 (11/2015)

Bitergia Inc. (2013): Report on the Activity of Companies in the WebKit Project.
http://blog.bitergia.com/2013/02/06/report-on-the-activity-of-companies-in-the-webkit-project/ (11/2015)

Black Duck Software Inc.; North Bridge Venture Partners (2015): The Future of Open Source 2014.
http://www.northbridge.com/2014-future-open-source-survey-results-0 (11/2015)

Blau, Peter M.; Scott, Richard W. (1962): *Formal Organizations. A Comparative Approach*. San Francisco: Chandler.

Boehm, Barry (1981): *Software Engineering Economics*. Englewood Cliffs: Prentice-Hall.

Bogers, Marcel (2012): Intellectual Property and Licensing Strategies in Open Collaborative Innovation. In: Pablos Heredero, Carmen de; Berzosa, David L.

(Hrsg.): *Open Innovation at Firms and Public Administrations. Technologies for Value Creation*. Hershey: IGI Global, S. 37–58.

Brooks, Frederick (1975): *The Mythical Man-Month.* Reading: Addison-Wesley.

Bulajewski, Mike (2011): The Peer Production Illusion, Part I. In: *MrTeaCup* vom 19.11.2011. http://www.mrteacup.org/post/peer-production-illusion-part-1.html (11/2015)

Burton, Grad (2002): A Personal Recollection: IBM's Unbundling of Software and Services. In: *IEEE Annals of the History of Computing* 24 (1): 64–71.

Byfield, Bruce (2013): What Makes for a Community Distribution? In*: Linux Magazine* vom 14.7.2013.
http://www.linux-magazine.com/What-makes-for-a-community-distribution (6/2015)

Cabrera, Angel; Cabrera, Elizabeth F. (2002): Knowledge-Sharing Dilemmas. In: *Organization Studies* 23 (5): 687–710.

Campbell-Kelly, Martin (2003): *From Airline Reservations to Sonic the Hedgehog. A History of the Software Industry*. Boston: MIT Press.

Canonical Group Ltd. (2014): Report and Financial Statements 2013.
http://de.scribd.com/doc/199373896/Canonical-Group-Limited-Annual-Accounts-2013 (11/2015)

Capek, Peter G.; Frank, Steven P.; Gerdt, Steve; Shields, David (2005): A History of IBM's Open-Source Involvement and Strategy. In: *IBM Systems Journal* 44 (2): 249–257.

Capiluppi, Andrea; Stol, Klaas-Jan; Boldyreff, Cornelia (2013): Exploring the Role of Commercial Stakeholders in Open Source Software Evolution. In: Hammouda, Imed; Lundell, Björn; Mikkonen, Tommi; Scacchi, Walt (Hrsg.): *Open Source Systems.* Heidelberg: Springer, S. 178–200.

Carhart, Richard (1953): *A Survey of the Current Status of the Electronic Reliability Problem*. Santa Monica: Rand Corporation.

Cerri, Davide; Fuggetta, Alfonso (2007): Open Standards, Open Formats, and Open Source. In: *Journal of Systems and Software* 80 (11): 1930–1937.

Ceruzzi, Paul E. (1998): *A History of Modern Computing*. Cambridge: MIT Press.

Claburn, Thomas (2007): Google's Secret Patent Portfolio Predicts gPhone. In: *InformationWeek* vom 19.9.2007.
http://www.informationweek.com/d/d-id/1059389 (11/2015)

Coleman, Gabriella E. (2013): *Coding Freedom. The Ethics and Aesthetics of Hacking*. Princeton: Princeton University Press.

Connell, Chuck (2000): Open Source Projects Manage Themselves? Dream On. IBM/Lotus Developers Network.
http://www.chc-3.com/pub/manage_themselves.htm (11/2015)

Corbet, Jonathan (2015): Development Activity in LibreOffice and OpenOffice. In: *LWN.net* vom 25.3.2015. https://lwn.net/Articles/637735/ (11/2015)

Corbet, Jonathan; Kroah-Hartman, Greg; McPherson, Amanda (2009–2015): Linux Kernel Development Report. San Francisco: The Linux Foundation.

Core Infrastructure Initiative (2015): Mission Statement. http://www.linuxfoundation.org/programs/core-infrastructure-initiative (11/2015)

Coté, Michael; Governor, James; O'Grady, Stephen (2007): Open Source Strategies. Analysis Paper. http://redmonk.com/public/goingopensource/Open-Source-Strategies.pdf (11/2015)

Cowan, Robin; Jonard, Nicolas (2003): The Dynamics of Collective Invention. In: *Journal of Economic Behavior & Organization* 52 (4): 513–532.

Cunningham, Jordan S. (2010): Interview: Arch Linux Team. In: *OS News* vom 11.1.2010. http://www.osnews.com/story/22692/Arch_Linux_Team (11/2015)

Dahlander, Linus; Magnusson, Mats (2008): How do Firms Make Use of Open Source Communities? In: *Long Range Planning* 41 (6): 629–649.

Debian (2015): Verfassung für das Debian-Projekt 1.5 (ratifiziert am 9.1.2015). https://www.debian.org/devel/constitution (11/2015)

Deshpande, Amit; Riehle, Dirk (2008): The Total Growth of Open Source. In: Russo, Barbara et al. (Hrsg.): *Open Source Development, Communities and Quality*. New York: Springer, S. 197–209.

Desmond, John P. (2001): Guarding the Citadel: IBM's Open Source Stance. In: *Software Magazine* 6/2001, 4.

Di Tullio, Dany; Staples, Sandy D. (2013): The Governance and Control of Open Source Software Projects. In: *Journal of Management Information Systems* 30 (3): 49–80.

Dickel, Sascha; Schrape, Jan-Felix (2015): Dezentralisierung, Demokratisierung, Emanzipation. Zur Architektur des digitalen Technikutopismus. In: *Leviathan* 43 (3): 442–463.

DistroWatch (2015): DistroWatch Ranking. http://distrowatch.com (11/2015)

Dolata, Ulrich (2015): Volatile Monopole. Konzentration, Konkurrenz und Innovationsstrategien der Internetkonzerne. In: *Berliner Journal für Soziologie* 24 (4): 505–529.

Dolata, Ulrich; Schrape, Jan-Felix (2015): Masses, Crowds, Communities, Movements: Collective Action in the Internet Age. In: *Social Movement Studies* (online veröffentlicht am 9. Juli 2015). Doi: 10.1080/14742837.2015.1055722

Driscoll, Kevin (2015): Professional Work for Nothing: Software Commercialization and 'An Open Letter to Hobbyists'. In: *Information & Culture* 50 (2): 257–283.

Driver, Mark (2013): Open Source is Everywhere in Modern IT. Präsentation, POSSCON 2013. Videodokument.
https://www.youtube.com/watch?v=cVWcegd_ce (11/2015)

Driver, Mark (2014): Within the Enterprise, Open Source Must Coexist in a Hybrid IT Portfolio. Gartner Inc. Research Report vom 23.8.2014. Stamford: Gartner.

Eclipse Foundation (2015a): 2015 Annual Eclipse Community Report.
https://eclipse.org/org/foundation/reports/annual_report.php (11/2015)

Eclipse Foundation (2015b): Eclipse Development Process 2014.
https://eclipse.org/projects/dev_process/ (11/2015)

Eitzen, Christopher von (2013): Q&A: Clement Lefebvre.
http://www.networkworld.com/article/2170903/software/q-a--clement-lefebvre--the-man-behind-linux-mint.html (11/2015)

Epstein, Richard A. (2015): Political Economy of Crowdsourcing: Markets for Labor, Rewards, and Securities. In: *The University of Chicago Law Review* 82: 35–52.

Europäische Kommission (2015): Kommission leitet förmliche Untersuchung zum mobilen Betriebssystem Android gegen Google ein. European Commission MEMO 15/4782.

Fisher, Franklin M.; McKie, James W.; Mancke, Richard B. (1983): *IBM and the US Data Processing Industry: An Economic History*. Santa Barbara: Praeger.

Fitzgerald, Brian (2006): The Transformation of Open Source Software. In: *MIS Quarterly* 30 (3): 587–598.

Fjeldstad, Øystein D.; Snow, Charles C.; Miles, Raymond E.; Lettl, Christopher (2012): The Architecture of Collaboration. In: *Strategic Management Journal* 33 (6): 734–750.

Forrester Research (2013): Market Update: Office 2013 and Productivity Suite Alternatives. Market Report. Cambridge: Forrester Inc.

Forrester Research (2014): Enterprise And SMB Software Survey, North America and Europe. Market Report. Cambridge: Forrester Inc.

Free Software Foundation (1986): *GNU's Bulletin* 1 (1).
https://www.gnu.org/bulletins/bull1.txt (11/2015)

Free Software Foundation (1989): GNU General Public License (GPL), Version 1.0.
http://www.gnu.org/licenses/old-licenses/gpl-1.0.html (11/2015)

Free Software Foundation (2013): Form 990.
http://static.fsf.org/nosvn/Form990_FY2013.pdf (11/2015).

Fuggetta, Alfonso (2003): Open Source Software – An Evaluation. In: *Journal of Systems and Software* 66 (1): 77–90.

Gartner Inc. (2015): Market Share: All Software Markets, Worldwide, 2014. Stamford: Gartner Inc.

Gates, Bill (1976): An Open Letter to Hobbyists. In: *Computer Notes* 1 (9): 3.

Gelsi, Steve (1999): VA Linux Rockets 698%. In: *CBS Marketwatch* vom 10.12.1999. http://www.cbsnews.com/news/va-linux-rockets-698/ (11/2015)

Germain, Jack M. (2014): Adobe's Open Source Tightrope Walk. In: *Linux Insider* vom 19.4.2014. http://www.linuxinsider.com/story/80329.html (11/2015)

Ghosh, Rishab Aiyer et al. (2006): *Economic Impact of Open Source Software on Innovation and the Competitiveness of the Information and Communication Technologies (ICT) Sector in the EU*. Maastricht: UNU-MERIT.

Gillen, Al (2013): Open Source Software. The Foundation for Tomorrow's Infrastructure. IDC Präsentation, Nagios World Conference 2013.

GNOME Foundation (2013): GNOME Annual Report 2013. https://www.gnome.org//wp-content/uploads/2014/09/GNOME-Annual-Report-2013.pdf (11/2015)

Gonsalves, Antone (2013): IBM Makes OpenStack the Cloud Platform to Beat. In: *ReadWrite* 5.3.2013. http://readwrite.com/2013/03/05/ibm-makes-openstack-the-cloud-platform-to-beat (11/2015)

Gonzalez-Barahona, Jesus M.; Izquierdo-Cortazar, Daniel; Maffulli, Stefano (2013): Understanding How Companies Interact with Free Software Communities. In: *IEEE Software* 30 (5): 38–45.

Gonzalez-Barahona, Jesus M.; Robles, Gregorio (2013): Trends in Free, Libre, Open Source Software Communities: From Volunteers to Companies. In: *it – Information Technology* 55 (5): 173–180.

Google Inc. (2010–2015a): Annual Report. Form 10-K. https://investor.google.com (11/2015)

Google Inc. (2015b): Android Corporate Contributor Agreement. http://source.android.com/source/cla-corporate.html (11/2015)

Google Inc. (2015c): Android Compatibility Definition. http://static.googleusercontent.com/media/source.android.com/de//compatibility/android-cdd.pdf (11/2015)

Greenstein, Shane; Nagle, Frank (2014): Digital Dark Matter and the Economic Contribution of Apache. In: *Research Policy* 43: 623–631.

Grüner, Sebastian (2015): OpenStack Foundation: „Wir werden von der Community überrannt". In: *Golem* vom 18.6.2015. http://www.golem.de/news/openstack-foundation-wir-werden-von-der-community-ueberrannt-1506-114728.html (11/2015)

Gulley, Ned; Lakhani, Karim (2010): The Determinants of Individual Performance and Collective Value in Private-collective Software Innovation. Harvard Business School Technology & Operations Management Unit Working Paper 10/065.

Hammond, Jeffrey (2014): Open Source by the Numbers. Präsentation, All Things Open Conference 2014. Videodokument.
https://www.youtube.com/watch?v=GTFM39h5m5g (11/2015)

Hardy, Quentin; Bilton, Nick (2014): Personality and Change Inflamed Mozilla Crisis. In: *New York Times* vom 4.4.2014, B1.

Henkel, Joachim; Schöberl, Simone; Alexy, Oliver (2014): The Emergence of Openness: How and Why Firms Adopt Selective Revealing in Open Innovation. In: *Research Policy* 43 (5): 879–890.

Herstatt, Cornelius; Ehls, Daniel (2015): *Open Source Innovation: Phenomenon, Participant Behaviour, Business Implications*. New York: Routledge.

Hogan, Mike (2009): The Corporate Closing of Open Source. In: *Planet MySQL* vom 22.5.2009. http://planet.mysql.com/entry/?id=19610 (11/2015)

Holtgrewe, Ursula; Werle, Raymund (2001): De-Commodifying Software? Open Source Software Between Business Strategy and Social Movement. In: *Science Studies* 14 (2): 43–65.

Hortonworks Inc. (2015): Financial Results for Fiscal Year 2014.
http://de.hortonworks.com/press-releases/hortonworks-reports-financial-results-fourth-quarter-fiscal-year-2014/ (11/2015)

IBM Corp. (1959): IBM 650 Model 4 Announcement. Press Release.
http://www-03.ibm.com/ibm/history/exhibits/650/650_pr4.html (11/2015)

IBM Corp. (2013): IBM Commits $1 Billion to Fuel Linux and Open Source Innovation on Power Systems. Press Release.
http://www-03.ibm.com/press/us/en/pressrelease/41926.wss (11/2015)

IBM Corp. (2015a): 2014 Annual Report.
http://www.ibm.com/annualreport/2014/bin/assets/IBM-Annual-Report-2014.pdf (11/2015)

IBM Corp. (2015b): Open Source and Standards.
http://www-03.ibm.com/linux/ossstds/ (11/2015)

International Data Corporation (2012): Worldwide Software Forecast.
http://s3.amazonaws.com/zanran_storage/www.bloomberg.com/ContentPages/2566616115.pdf (11/2015)

International Data Corporation (2014): Worldwide Server Market Revenues Decline –4.4% in Q4. Press Release.
http://www.idc.com/getdoc.jsp?containerId=prUS24704714 (11/2015)

Jaeger, Till (2010): Enforcement of the GNU GPL in Germany and Europe. In: *Journal of Intellectual Property, Information Technology and E-Commerce Law* 1 (1): 34–39.

Jive Software Inc. (2014): Form 10-K. http://investors.jivesoftware.com (11/2015)

Johnson, Luanne (2002): A View from the Sixties: How the Software Industry Began. In: Akera, Atsushi; Nebeker, Fredrik (Hrsg.): *From 0 to 1. An Authoritative History of Modern Computing*. New York: Oxford University Press, S. 101–109.

Juliussen, Karen; Juliussen, Egil (1990): *The Computer Industry Almanac 1991*. New York: Brady.

KDE e.V. (2013): Community Report 27. https://ev.kde.org/reports (11/2015)

KDE Project (2015): Project Management.
https://www.kde.org/community/whatiskde/ (11/2015)

Kernel.Org Documentation (2015a): How the Development Process Works.
https://www.kernel.org/doc/Documentation/development-process/2.Process (11/2015)

Kernel.Org Documentation (2015b): How to Get Your Change Into the Linux Kernel.
https://www.kernel.org/doc/Documentation/SubmittingPatches (11/2015)

Kolassa, Carsten; Riehle, Dirk; Riemer, Philip; Schmidt, Michael (2014): Paid vs. Volunteer Work in Open Source. In: *Proceedings 47th Hawaii Int. Conference on System Sciences*, S. 3286–3295.

Kostakis, Vasilis; Papachristou, Marios (2014): Commons-based Peer Production and Digital Fabrication. In: *Telematics and Informatics* 31 (3): 434–443.

Krafft, Martin F. (2010): *A Delphi Study of the Influences on Innovation Adoption and Process Evolution in a Large Open Source Project: The Case of Debian*. Dissertation. Limerick: University of Limerick, Department of Computer Science & Information Systems.

Kreiss, Daniel; Finn, Megan; Turner, Fred (2011): The Limits of Peer Production: Some Reminders from Max Weber for the Network Society. In: *New Media & Society* 13 (2): 243–259.

Lakhani, Karim R.; Hippel, Eric von (2003): How Open Source Software Works. In: *Research Policy* 32 (6): 923–943.

Larsson, Rikard; Bengtsson, Lars; Henriksson, Kristina; Sparks, Judith (1998): The Interorganizational Learning Dilemma: Collective Knowledge Development in Strategic Alliances. In: *Organization Science* 9 (3): 285–305.

Leonard, Andrew (2001): Life, Liberty and the Pursuit of Free Software. In: *Salon* vom 16.2.2001. http://www.salon.com/2001/02/15/unamerican/ (11/2015)

Lerner, Joshua (2012): *The Architecture of Innovation*. Boston: Harvard Business Press.

Lerner, Joshua; Schankerman, Mark (2010): *The Comingled Code. Open Source and Economic Development*. Cambridge: MIT Press.

Lerner, Joshua; Tirole, Jean (2002): Some Simple Economics of Open Source. In: *Journal of Industrial Economics* 50 (2): 197–234.

Lerner, Joshua; Tirole, Jean (2005): The Scope of Open Source Licensing. In: *Journal of Law, Economics & Organization* 21 (1): 20–56.

Lessig, Lawrence (1999): Open Code and Open Societies: Values of Internet Governance. In: *Chicago Kent Law Review* 74: 1405–1420.

Levy, Steven (1984): *Hackers: Heroes of the Computer Revolution*. Garden City: Anchor Press.

Lewine, Peter (2014): Why There Will Never Be Another Red Hat: The Economics Of Open Source. In: *Techcrunch* vom 13.2.2014. http://on.tcrn.ch/l/pjXf (11/2015)

Li, Yan; Tan, Chuan; Teo, Hock-Hai (2012): Leadership Characteristics and Developers' Motivation in Open Source Software Development. In: *Information & Management* 49 (5): 257–267.

Lowood, Henry (2009): Videogames in Computer Space: The Complex History of Pong. In: *IEEE Annals of the History of Computing* 31 (3): 5–19.

March, James; Simon, Herbert (1958): *Organizations*. Cambridge: Blackwell.

McAllister, Neil (2013): Microsoft no longer a Top Kernel Contributor. In: *The Register* 16.9.2013.
http://www.theregister.co.uk/2013/09/16/linux_foundation_kernel_report_2013/ (11/2015)

McGaw, Judith A. (1987): *Most Wonderful Machine: Mechanization and Social Change in Berkshire Paper Making 1801–1885*. Princeton: Princeton University Press.

McHugh, Josh (1998): For the Love of Hacking. In: *Forbes* vom 10.8.1998.
http://www.forbes.com/forbes/1998/0810/6203094a.html (11/2015)

Menell, Peter S. (2002): Envisioning Copyright Law's Digital Future. In: *New York Law School Review* 46: 63–199.

Meyer, Peter B. (2003): Episodes of Collective Invention. BLS Working Paper 368. Washington: U.S. Bureau of Labor Statistics.

Microsoft Corp. (2015a): 2014 Annual Report.
http://www.microsoft.com/investor/reports/ (11/2015).

Microsoft Corp. (2015b): Openness.
http://www.microsoft.com/en-us/openness/index.aspx (11/2015).

Moody, Glyn (2002): *Rebel Code. The Inside Story of Linux and the Open Source Revolution*. New York: Basic Books.

Mowery, David C. (1999): The Computer Software Industry. In: Mowery, David C.; Nelson, Richard W. (Hrsg.): *Sources of Industrial Leadership.* Cambridge: Cambridge University Press, S. 133–168.

Mowery, David C.; Langlois, Richard N. (1996): Spinning off and Spinning on(?): The Federal Government Role in the Development of the US Computer Software Industry. In: *Research Policy* 25 (6): 947–966.

Mozilla Foundation (2014): Independent Auditors' Report and Consolidated Financial Statements.
https://static.mozilla.com/moco/en-US/pdf/Mozilla_Audited_Financials_2013.pdf (11/2015)

Mozilla Foundation (2015a): Mozilla Organizations.
https://www.mozilla.org/en-US/about/ (11/2015)

Mozilla Foundation (2015b): Mozilla Roles and Leadership.
https://www.mozilla.org/en-US/about/governance/roles/ (11/2015)

Mulligan, Catherine (2013): *The Communications Industries in the Era of Convergence*. London: Routledge.

NetApplications Inc. (2015): Desktop Operating System Market Share.
http://www.netmarketshare.com/operating-system-market-share.aspx (11/2015)

Netcraft Ltd. (2015): September 2015 Web Server Survey.
http://news.netcraft.com/archives/2015/09/16/september-2015-web-server-survey.html (11/2015)

Netscape Communications Corp. (1998): Netscape Announces mozilla.org. Press Release 23.2.1998.

Nuvolari, Alessandro (2004): Collective Invention during the British Industrial Revolution: The Case of the Cornish Pumping Engine. In: *Cambridge Journal of Economics* 28 (3): 347–363.

O'Mahony, Siobhán (2003): Guarding the Commons. How Community Managed Software Projects Protect their Work. In: *Research Policy* 32 (7): 1179–1198.

O'Mahony, Siobhán (2005): Non-profit Foundations and their Role in Community-Firm Software Collaboration. In: Feller, Joseph; Fitzgerald, Brian; Hissam, Scott A.; Lakhani, Karim R. (Hrsg.): *Perspectives on Free and Open Source Software*. Cambridge: MIT Press, S. 23–46.

O'Mahony, Siobhán; Ferraro, Fabrizio (2007): The Emergence of Governance in an Open Source Community. In: *Academy of Management Journal* 50 (5): 1079 bis 1106.

Open Hub (2015): Project Summaries, Contributors Listings.
https://www.openhub.net/ (11/2015)

Open Source Matters Inc. (2015): Budget vs. Actuals – FY 2014.
http://opensourcematters.org/images/financials/2014/Budget_vs_Actuals__FY2014-12.pdf (11/2015)

Osterloh, Margit; Rota, Sandra (2007): Open Source Software Development: Just Another Case of Collective Invention? In: *Research Policy* 36 (2): 157–171.

Perens, Bruce (1999): The Open Source Definition. In: DiBona, Chris; Ockman, Sam; Stone, Mark (Hrsg.): *Opensources. Voices from the Open Source Revolution*. Sebastopol: O'Reilly, S. 171–188.

Perlroth, Nicole (2014): Heartbleed Highlights a Contradiction in the Web. In: *The New York Times* vom 18.4.2014. http://nyti.ms/1hb6uBd (11/2015)

Perrier, Christian (2014): Developers per Country. In: *Bubulle's Weblog* vom 29.7.2014.
http://www.perrier.eu.org/weblog/2014/07/29#devel-countries-201308 (11/2015)

Phister, Montgomery (1976): *Data Processing. Technology and Economics.* Bedford: Santa Monica.

Phlow Magazine (2003): Typo3 und Kasper Skårhø – Was Open Source und Christentum gemeinsam haben. In: *Phlow* vom 29.4.2003.
http://phlow.net/mag/interview_portrait/typo3_und_kasper_skrh_was_open_source_und_christentum_gemeinsam_haben.php (1/2016)

PriceWaterhouseCoopers AG (2014): PwC Global 100 Software Leaders. London: PWC.

Raymond, Eric S. (1998a): The Revenge of the Hackers. In: DiBona, Chris; Ockman, Sam; Stone, Mark (Hrsg.): *Opensources. Voices from the Open Source Revolution*. Sebastopol: O'Reilly, S. 207–220.

Raymond, Eric S. (1998b): Goodbye, "free software"; Hello, "open source". Announcement vom 22.11.1998.
ftp://ftp.lab.unb.br/pub/computing/museum/esr/open-source.html (11/2015)

Raymond, Eric S. (1999): *The Cathedral and the Bazaar. Musings on Linux and Open Source by an Accidental Revolutionary*. Sebastopol: O'Reilly.

Raymond, Eric S. (2000): Project Structures and Ownership. In: Homesteading the Noosphere (Version 3.0).
http://catb.org/~esr/writings/homesteading/homesteading/ar01s16.html (11/2015)

Reimer, Jeremy (2012): From Altair to iPad: 35 Years of Personal Computer Market Share. In: *Ars Technica* vom 14.8.2012. http://bit.ly/OwzHKm (11/2015)

Riehle, Dirk (2012): The Single-Vendor Commercial Open Source Business Model. In: *Information Systems and E-Business Management* 10 (1): 5–17.

Riehle, Dirk (2013): The Unstoppable Rise of Open Source. In: *Information Technology* 55 (5): 171 f.

Rifkin, Jeremy (2014): *The Zero Marginal Cost Society: The Internet of Things, the Collaborative Commons, and the Eclipse of Capitalism*. New York: Palgrave Macmillan.

Rivlin, Gary (2003): Leader of the Free World: How Linus Torvalds Became Benevolent Dictator of Planet Linux. In: *Wired* 11 (11): 157.

Roeckx, Kurt (2015): Debian Project Leader Election 2015 Results. https://lists.debian.org/debian-devel-announce/2015/04/msg00003.html (11/2015)

Romer, Paul (1990): Endogenous Technological Change. In: *Journal of Political Economy* 98 (5): 71–102.

Sands, Rich (2012): Measuring Project Activity. In: *Back Duck Open Hub Blog* vom 10.4.2012. http://blog.openhub.net/2012/04/measuring-project-activity/ (11/2015)

Sayadian, Helga (1990): *The Information Technology Industry Data Book 1960–2000*. Washington: Computer and Business Equipment Manufacturers Association.

Schaarschmidt, Mario; Walsh, Gianfranco; Kortzfleisch, Harald F. (2015): How Do Firms Influence Open Source Software Communities? In: *Information and Organization* 25 (2): 99–114.

Schneier, Bruce (2014): Heartbleed. In: *Schneier on Security* vom 9.4.2014. https://www.schneier.com/blog/archives/2014/04/heartbleed.html (11/2015)

Schrape, Jan-Felix (2012): *Wiederkehrende Erwartungen. Prognosen, Visionen und Mythen um neue Medien seit 1970*. Boizenburg: Hülsbusch.

Schwarz, Michael; Takhteyev, Yuri (2010): Half a Century of Public Software Institutions. In: *Journal of Public Economic Theory* 12 (4): 609–639.

Singer, Harold (1976): An Open Letter to Ed Roberts. In: *Micro-8 Computer User Newsletter* 2 (4): 1.

Snow, Shane (2014): How Matt's Machine Works. In: *Fast Company* vom 11.9.2014. http://www.fastcompany.com/3035463/how-matts-machine-works (11/2015)

Software in the Public Interest Inc. (2015): 2015 Annual Report. http://www.spi-inc.org/corporate/annual-reports/2015.pdf (11/2015)

Spencer, Jennifer W. (2003): Firms' Knowledge-sharing Strategies in the Global Innovation System. Evidence from the Flat Panel Display Industry. In: *Strategic Management Journal* 24 (3): 217–233.

Spiegel (o.V.) (1983): Computer – das ist wie eine Sucht. In: *Der Spiegel* 50/1983: 172–183.

Spiegel (o.V.) (1999): Konkurrenz für Microsoft. In: *Der Spiegel* 33/1999: 78.

Spiegel (o.V.) (2000): Aufrüstung im Patentkrieg. In: *Der Spiegel* 30/2000: 54–58.

Spies, Rüdiger (2010): SAP und Open Source. In: *IT-Business* vom 14.12.2010. http://www.it-business.de/partnerzones/idc-research-zone/viewpoints/articles/296134/ (11/2015)

Spreeuwenberg, Kimberley; Poell, Thomas (2012): Android and the Political Economy of the Mobile Internet. In: *First Monday* 17 (7). http://dx.doi.org/10.5210/fm.v17i7.4050 (11/2015)

Stallman, Richard (1983): New UNIX Implementation. http://bit.ly/1DSDoXW (11/2015)

Stallman, Richard (1985): GNU Manifesto. In: *Dr. Dobb's Journal of Software Tools* 10 (3): 30.

Stallman, Richard (2002): *Free Software, Free Society.* Boston: GNU Press.

Stamelos, Ionnais (2014): Management and Coordination of Free/Open Source Projects. In: Ruhe, Günther; Wohlin, Claes (Hrsg.): *Software Projekt Management in a Changing World*. New York: Springer, S. 321–341.

StatCounter Inc. (2015): Global Stats. http://gs.statcounter.com/ (11/2015)

Steinmueller, William E. (1995): The US Software Industry. An Analysis and Interpretive History. In: Mowery, David C. (Hrsg.): *The International Computer Software Industry. A Comparative Study of Industry Evolution and Structure*. New York: Oxford University Press, S. 15–52.

Stiller, Ase (2011): The Open Source Trials: Hanging in the Legal Balance of Copyright and Copyleft. In: *Vision Mobile Blog* vom 15.3.2011. http://www.visionmobile.com/blog/2011/03/the-open-source-trials-hanging-in-the-legal-balance-of-copyright-and-copyleft/ (11/2015)

Stokel-Walker, Chris (2014): The Internet Is Being Protected By Two Guys Named Steve. In: *Buzzfeed* vom 25.4.2014. http://www.buzzfeed.com/chrisstokelwalker/the-internet-is-being-protected-by-two-guys-named-st#.ooP0j2q2z (11/2015)

Suddaby, Roy (2013): Book Review: The Janus Face of Commercial Open Source Software Communities. In: *Organization Studies* 34 (7): 1009–1011.

Tapscott, Don; Williams, Anthony D. (2006): *Wikinomics. How Mass Collaboration Changes Everything*. New York: Portfolio.

Teixeira, Jose; Lin, Tingting (2014): Collaboration in the Open-Source Arena: The WebKit Case. In: *Proceedings of the 52nd ACM Conference on Computers and People*. New York: ACM, S. 121–129.

The Document Foundation (2015): 2014 Annual Report. https://wiki.documentfoundation.org/ File:TDF2014AnnualReport.pdf (11/2015)

Time Magazine (o.V.) (1978): The Computer Society: Pushbutton Power. In: *Time Magazine* 111 (9): 46–49.

Torvalds, Linus (1998): LINUX Manifesto. Interview. In: *boot Magazine* (July/Aug.): 32–37.

Torvalds, Linus (2002): Re: [PATCH] Remove Bitkeeper Documentation from Linux Tree. In: *Linux Kernel Mailinglist* vom 20.4.2002.
http://lwn.net/2002/0425/a/ideology-sucks.php3 (11/2015)

Trendowicz, Adam; Jeffery, Ross (2014): Constructive Cost Model—COCOMO. In: dies.: *Software Project Effort Estimation*. Heidelberg/New York: Springer, S. 277–293.

Tukey, John W. (1958): The Teaching of Concrete Mathematics. In: *American Mathematical Monthly* 65 (1): 1–9.

Typo3 Association (2015): Financial Statements for the Year 2014.
http://typo3.org/fileadmin/t3org/images/FM-content/association/financial/Financial_Statements_2014.pdf (11/2015)

Ubuntu Project (2015): Governance.
http://www.ubuntu.com/about/about-ubuntu (11/2015)

UNCTAD – United Nations Conference of Trade and Development (2012): *Information Economy Report 2012*. New York/Genf: United Nations.

Vision Mobile Ltd. (2011): Open Governance Index. London: Vision Mobile.

W3techs Surveys (2015): Technologies Overview.
http://w3techs.com/technologies (11/2015)

Walli, Stephen; Gynn, Dave; Rotz, Bruno von (2005): The Growth of Open Source Software in Organizations. Business Report. Boston: Optaros.

Warren, Jim C. (1976): Correspondence. In: *SIGPLAN Notices* 11 (7): 1 f.

Wasserman, Anthony (2013): Community and Commercial Strategies in Open Source Software. In: *Information Technology* 55 (5): 181–188.

Watson, Richard T.; Boudreau, Marie-Claude; York, Paul T.; Greiner, Martina E.; Wynn, Donald (2008): The Business of Open Source. In: *Communications of the ACM* 51 (4): 41–46.

Weber, Steven (2005): *The Success of Open Source*. Cambridge: Harvard University Press.

Werle, Raymund (2000): *Institutional Aspects of Standardization. Jurisdictional Conflicts and the Choice of Standardization Organizations*. MPIfG Discussion Paper 2000/1. Köln: MPIfG.

West, Joel; Bogers, Marcel (2014): Leveraging External Sources of Innovation. A Review of Research on Open Innovation. In: *Journal of Product Innovation Management* 31 (4): 814–831.

West, Joel; Dedrick, Jason (2001): Open Source Standardization: The Rise of Linux in the Network Era. In: *Knowledge, Technology & Policy* 14 (2): 88–112.

West, Joel; Salter, Ammon; Vanhaverbeke, Wim; Chesbrough, Henry (2014): Open Innovation: The Next Decade. In: *Research Policy* 43 (5): 805–811.

Westenholz, Ann (2007): Cannibalizing Your Own Business? Commercializing the Open Source Software TYPO3. TII Working Paper Series, Vol. 20. Edmonton: University of Alberta.
https://business.ualberta.ca/en/Centres/TCC/~/media/business/Centres/TCC/Documents/WorkingPapers/TII20Westenholz.ashx (1/2016)

Westenholz, Ann (Hrsg.) (2012): *The Janus Face of Commercial Open Source Software Communities.* Copenhagen: Copenhagen Business School Press.

Register

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z

Weitere Titel aus dem vwh-Verlag (Auszug)

Reihe „Medienwirtschaft“

C. Frahm: Die Zukunft der Tonträgerindustrie 2007, 24,90 €, 978-3-9802643-8-9

S. Huber: Neue Erlösmodelle für Zeitungsverlage 2007, 27,90 €, ISBN 978-3-9802643-9-6

K. Huemer: Die Zukunft des Buchmarktes Verlage und Buchhandlungen im digitalen Zeitalter 2010, 24,90 €, ISBN 978-3-940317-73-5

J.-F. Schrape: Gutenberg-Galaxis Reloaded? Der Wandel des deutschen Buchhandels durch Internet, E-Books und Mobile Devices 2011, 17,90 €, ISBN 978-3-940317-85-8

B. Blaha: Von Riesen und Zwergen Zum Strukturwandel im verbreitenden Buchhandel in Deutschland und Österreich 2011, UVP 9,80 €, ISBN 978-3-940317-93-3

J. Stiglhuber: Macht und Ohnmacht der Unabhängigen Independent-Verlage und ihre Verortung 2011, 26,90 €, ISBN 978-3-86488-003-2

K. Werneburg: Print Is ‚Easy‘, Online Is ‚Tough‘ Eine psychophysiologische Untersuchung zur mentalen Beanspruchung durch crossmediale Angebote 2016, 33,80 €, ISBN 978-3-86488-094-0

Reihe „E-Business“

S. Sobczak/M. Groß: Crowdsourcing 2010, 24,90 €, ISBN 978-3-940317-61-2

C. Noack: Crossmedia Marketing 2010, UVP 13,80 €, 978-3-940317-78-0

Reihe „E-Learning“

A. S. Nikolopoulos: Sicherung der Nachhaltigkeit von E-Learning-Angeboten an Hochschulen 2010, 32,50 €, ISBN 978-3-940317-60-5

C. Biel: Personal Learning Environments als Methode zur Förderung des selbstorganisierten Lernens 2011, 24,90 €, ISBN 978-3-86488-001-8

K. Himpsl-Gutermann: E-Portfolios in der universitären Weiterbildung 2012, 30,90 €, ISBN 978-3-86488-014-8

M. Beißwenger/N. Anskeit/A. Storrer (Hg.): Wikis in Schule und Hochschule 2012, 36,90 €, ISBN 978-3-86488-017-9

C. Lehr: Web 2.0 in der universitären Lehre 2012, 27,90 €, 978-3-86488-024-7

J. Wagner/V. Heckmann (Hg.): Web 2.0 im Fremdsprachenunterricht Ein Praxisbuch für Lehrende in Schule und Hochschule 2012, 27,50 €, ISBN 978-3-86488-022-3

E. Blaschitz et al. (Hg.): Zukunft des Lernens Wie digitale Medien Schule, Aus- und Weiterbildung verändern 2012, 23,50 €, ISBN 978-3-86488-028-5

U. Höbarth: Konstruktivistisches Lernen mit Moodle Praktische Einsatzmöglichkeiten in Bildungsinstitutionen *- 3. Aufl. -* 2013, 31,50 €, ISBN 978-3-86488-033-9

A. Klampfer: E-Portfolios als Instrument zur Professionalisierung in der Lehrer- und Lehrerinnenausbildung 2013, 27,90 €, ISBN 978-3-86488-034-6

C. Koenig: Bildung im Netz Analyse und bildungstheoretische Interpretation der neuen kollaborativen Praktiken in offenen Online-Communities 2013, 31,90 €, ISBN 978-3-86488-042-1

B. Getto: Anreize für E-Learning Eine Untersuchung zur nachhaltigen Verankerung von Lerninnovationen an Hochschulen 2013, 26,90 €, ISBN 978-3-86488-052-0

M. Gecius: Game-based Learning in der Schule *Hamlet* als Computerspiel – Die Einsatzmöglichkeiten der computerbasierten Spielform des Adventures im literaturbezogenen Englischunterricht 2014, 37,50 €, ISBN 978-3-86488-066-7

M. Kirchner: Social-Software-Portfolios im Einsatz Zwischen Online-Lernen und Medienkompetenz im selbstgesteuert-konnektiven Lernalltag 2015, 32,90 €, ISBN 978-3-86488-075-9

Reihe „Web 2.0“

J. Moskaliuk (Hg.): Konstruktion und Kommunikation von Wissen mit Wikis 2008, 27,50 €, ISBN 978-3-940317-29-2

H. Frohner: Social Tagging 2010, 26,90 €, ISBN 978-3-940317-03-2

R. Bauer: Die digitale Bibliothek von Babel Über den Umgang mit Wissensressourcen im Web 2.0 2010, 26,90 €, ISBN 978-3-940317-71-1

J. Jochem: Performance 2.0 Zur Mediengeschichte der Flashmobs 2011, 24,90 €, ISBN 978-3-940317-98-8

G. Franz: Die vielen Wikipedias
Vielsprachigkeit als Zugang zu einer globalisierten Online-Welt
2011, 27,50 €, ISBN 978-3-86488-002-5

R. Sonnberger: Facebook im Kontext medialer Umbrüche
2012, 29,50 €, ISBN 978-3-86488-009-4

J. Brailovskaia: Narzisstisch und sensationssuchend?
Eine Studie zum Zusammenhang zwischen Persönlichkeitsmerkmalen und Online-Selbstdarstellung am Beispiel von *studiVZ*
2013, 24,50 €, ISBN 978-3-86488-039-1

C. Kaiser: Soziale Schließung online
Bildung, Netzwerke und Strategien in *XING*
2015, 26,90 €, ISBN 978-3-86488-088-9

Reihe „Game Studies“

B. Sterbenz: Genres in Computerspielen – eine Annäherung
2011, 24,50 €, ISBN 978-3-940317-99-5

D. Appel et al. (Hg.): Welt|Kriegs|Shooter
Computerspiele als realistische Erinnerungsmedien? 2012, 28,50 €, 978-3-86488-010-0

S. Felzmann: Playing Yesterday
Mediennostalgie im Computerspiel
2012, 22,50 €, ISBN 978-3-86488-015-5

M. Breuer (Hg.): E-Sport – Perspektiven aus Wissenschaft und Wirtschaft
2012, 26,90 €, ISBN 978-3-86488-026-1

R. T. Inderst/P. Just (Hg.): Build ’em Up – Shoot ’em Down
Körperlichkeit in digitalen Spielen
2013, 35,90 €, ISBN 978-3-86488-027-8

J. Koubek/M. Mosel/S. Werning (Hg.): Spielkulturen Funktionen und Bedeutungen des Phänomens Spiel in der Gegenwartskultur und im Alltagsdiskurs
2013, 23,90 €, ISBN 978-3-86488-056-8

C. Huberts/S. Standke (Hg.): Zwischen|Welten Atmosphären im Computerspiel
2014, 34,90 €, ISBN 978-3-86488-063-6

B. Beil/M. Bonner/T. Hensel (Hg.): Computer|Spiel|Bilder
2014, 32,80 €, ISBN 978-3-86488-062-9

L. Cannellotto: Digitale Spiele und Hybridkultur
2014, 24,50 €, ISBN 978-3-86488-046-9

J.-M. Loebel: Lost in Translation
Leistungsfähigkeit, Einsatz und Grenzen von Emulatoren bei der Langzeitbewahrung digitaler multimedialer Objekte am Beispiel von Computerspielen
2014, 26,80 €, ISBN 978-3-86488-068-1

C. Nibler: Achievement & Exploration
Dramaturgie der Grenzüberschreitung im Computerspiel
2015, 35,80 €, ISBN 978-3-86488-079-7

A.-M. Letourneur/M. Mosel/T. Raupach (Hg.): Retro-Games und Retro-Gaming
Nostalgie als Phänomen einer performativen Ästhetik von Computer- und Videospielkulturen 2015, 30,80 €, 978-3-86488-078-0

Reihe „E-Humanities“

G. Vogl: Selbstständige Medienschaffende in der Netzwerkgesellschaft
2008, UVP 11,80 €, 978-3-940317-38-4

C. Russ: Online Crowds Massenphänomene und kollektives Verhalten im Internet
2010, 31,50 €, ISBN 978-3-940317-67-4

C. Potzner: Chancen und Risiken der Arbeit im E-Business
2010, UVP 12,80 €, 978-3-940317-70-4

M. Janneck/C. Adelberger: Komplexe Software-Einführungsprozesse gestalten: Grundlagen und Methoden Am Beispiel eines Campus-Management-Systems
2012, 26,90 €, ISBN 978-3-940317-63-6

H. Kohle: Digitale Bildwissenschaft
2013, 16,80 €, ISBN 978-3-86488-036-0

C. Krause/R. Reiche: Ein Bild sagt mehr als tausend Pixel? Digitale Forschungsansätze in den Bild- und Objektwissenschaften 2015, 19,80 €, ISBN 978-3-86488-076-6

Reihe „Typo | Druck“

M. Liebig: Browser-Typografie
2008, 35,90 €, ISBN 978-3-940317-09-4

D. Meletis: Graphetik
Form und Materialität von Schrift
2015, 28,50 €, ISBN 978-3-86488-087-2

weitere Schriftenreihen des vwh-Verlages (s. www.vwh-verlag.de):

· Medientheorie · Multimedia · AV-Medien · E-Collaboration · Schriften des Innovators Club · Schriften zur Informationswissenschaft · Kleine Schriften

vwh Aktuelle Ankündigungen, Inhaltsverzeichnisse und Rezensionen finden sie im vwh-Blog unter www.vwh-verlag.de.
Das komplette Verlagsprogramm mit Buchbeschreibungen sowie eine direkte Bestellmöglichkeit im vwh-Shop finden Sie unter www.vwh-verlag-shop.de.